

THÈSE

présentée à

L'UNIVERSITÉ de PARIS-DAUPHINE

pour obtenir le grade de

DOCTEUR EN SCIENCES

Spécialité

**MATHÉMATIQUES APPLIQUÉES ET TRAITEMENT
D'IMAGES**

par

Thomas DESCHAMPS

Sujet de la thèse :

**EXTRACTION DE COURBES ET SURFACES
PAR MÉTHODES DE CHEMINS MINIMAUX ET
ENSEMBLES DE NIVEAUX. APPLICATIONS
EN IMAGERIE MEDICALE 3D.**

Soutenue le 20 décembre 2001 à l'Université devant un jury composé de :

M	Olivier	FAUGERAS	Président
M	Philippe	CINQUIN	Rapporteurs
M	Ron	KIMMEL	
M	Laurent	COHEN	Directeur
Mme	Françoise	DIBOS	Examinateurs
M	Sherif	MAKRAM-EBEID	
M	Wiro	NIESSEN	
M	Nicolas	ROUGON	

A toi

(*J. Dassin, 1938-1980*)

Remerciements

Je tiens à remercier tout d'abord Messieurs Philippe Cinquin et Ron Kimmel pour avoir accepté le travail de rapporteurs de cette thèse. Monsieur Cinquin a été le premier à me faire entrevoir les possibles applications mathématiques au domaine médical, lors d'une présentation des activités de son laboratoire. En ce qui concerne Monsieur Kimmel

אני מודה מאוד את רון קימל על כך שעשה דרך כדי להשתתף בועידת הובנים

Madame Françoise Dibos, qui m'a il y a bien longtemps enseigné les principes fondamentaux du traitement du signal, a accepté de participer à ce jury. Je tiens à l'en remercier, tout comme Monsieur Olivier Faugeras, pionnier de la vision par ordinateur, dont l'influence s'étend aussi maintenant au domaine médicale.

Monsieur Nicolas Rougon, figure omniprésente du traitement d'image et des modèles déformables, me fait aussi l'honneur d'être membre du jury.

Ik zou in het bijzonder een dankwoord willen richten aan mijnheer Wiro Niessen, die helemaal uit het hoge noorden hierheen is gekomen. Het is voor mij een eer om een specialist inzake medische beeldverwerking in de jury te hebben.

Pour m'avoir guidé dans mes recherches tout au long de cette thèse, et pour m'avoir laissé aller dans les directions qui m'intéressaient, pour m'avoir appris à communiquer sur mon travail et encouragé à publier, je tiens à remercier Monsieur Laurent Cohen. Merci de m'avoir fait confiance, et de m'avoir poussé en avant.

Monsieur Sherif Makram-Ebeid me fait l'honneur et l'amitié d'être lui aussi membre du jury. Ses compétences autant théoriques que techniques ont toujours été pour moi digne du plus grand respect. Son dynamisme et son aisance avec l'ensemble des théories de la vision (et autres) resteront pour moi un modèle du genre.

Je souhaite remercier toute l'équipe du groupe MediSys de Philips Recherche France, pour m'avoir accueilli en son sein depuis déjà plus de trois ans. La participation des gens de cette équipe à mon travail ne se limite pas à l'enseignement que j'ai pu tirer de leurs compétences et aux nombreuses contributions de chacun à mon ouvrage. Au delà de la bonne humeur et de l'ambiance sympathique, j'ai vraiment apprécié de faire partie de ce groupe: Cyrille Allouche, Odile Bonnefous, Jacques Breitenstein, Antoine Collet-Billon, Claude Cohen-Bacrie, Eric Denis, Maxim Fradkin, Raoul Florent, Olivier Gérard, Laurence Germond, Yves Hily, Françoise Iritz, Marie Jacob, Jean-Michel Lagrange, Pierre Lelong, Claire Lévrier, Claude Méquio, Lucille Nosjean, Jean Pergrale, Jean-Michel Rouet, Michel Siméon, et Nicolas Villain. J'en profite pour remercier Sandra Delcorso pour m'avoir soutenu pendant ces derniers mois de travail.

Pour ceux qui ont quitté le groupe entre temps, je souhaiterais distribuer quelques remerciements, notamment à Jean-Michel Létang, qui a été mon “parrain” au sein de l’équipe, et a guidé mes premiers pas dans le domaine de la recherche industrielle, tout en me faisant bénéficier de sa rigueur et sa créativité. Une mention très spéciale revient tout naturellement à Xavier Merlhiot et aussi Myriam Greff, anciens stagiaires du groupe: travailler avec vous a été plus qu’un plaisir, et je ne peux que reconnaître l’influence primordiale de nos collaborations sur ma thèse, pour les “Ensembles de Niveaux” en ce qui concerne Xavier, et le “Live-Wire” pour Myriam. Vos compétences sont grandes et j’ai pris grand plaisir à travailler avec vous, et à inclure dans ma thèse le fruit de ces collaborations.

Au sein de la galaxie Philips, je tiens à remercier l’équipe MIMIT-Advanced Development de Frans Gerritsen, et en particulier Roel Truyen et Bert Verdonck, pour leur accueil chaleureux, et leur intérêt pour mon travail qui s’est trouvé concrétisé dans le développement de certains prototypes, et dans plusieurs publications communes.

Mes derniers remerciements vont tout naturellement à ceux qui m’ont soutenu pendant ces longues années d’études, c’est à dire à mes parents et toute ma famille. Merci aussi à tous mes amis qui ont su supporter les baisses de moral d’un chercheur souvent égaré voir perdu entre deux raisonnements. Vous êtes trop nombreux pour tous vous citer, mais mes remerciements s’adressent au clan de Boulogne-Billancourt, dans son sens le plus large. Je tiens à remercier Hélène Sellier pour m’avoir laissé mettre à contribution son grand talent pour le dessin et autorisé à mettre des représentations de ses œuvres au début de chacune de mes parties. Et enfin merci à ceux et celles qui ont su me donner la force et la motivation nécessaires pour terminer, qui ont su, du fait de leur soutien, de leur présence ou de leur simple existence, donner une signification et un but à tout cela.

Thomas Deschamps
Paris, Décembre 2001

Contents

Remerciements	v
Abbreviations	xi
Introduction	1
I Extraction de Chemins	5
1 Les Chemins Minimaux en Traitement d'Images	9
1.1 Minimal Paths theory	10
1.1.1 The minimal path in geometrical optic	10
1.1.2 Minimal path for curve extraction	11
1.2 The Cohen-Kimmel Method in 2D	12
1.2.1 Global minimum for Active Contours	12
1.2.2 Problem formulation	13
1.3 Numerical Implementation	14
1.3.1 Graph Search Algorithms and Metrication Error	14
1.3.2 Fast Marching Resolution	15
1.3.3 Back-propagation	17
1.3.4 Comparing Dijkstra and Eikonal	17
1.4 The Regularity of the Path	19
1.4.1 Influence on the gradient descent scheme	19
1.4.2 Influence on the number of points visited	20
2 Nouvelles Techniques d'Extraction de Chemins à l'Aide du Fast-Marching	23
2.1 3D Extension	24
2.2 Several minimal path extraction techniques	24
2.2.1 Partial Front Propagation	26
2.2.2 Simultaneous Front Propagation	27
2.2.3 Euclidean Path Length Computation	29
2.3 Path centering in linear objects	30
2.3.1 Segmentation step	31
2.3.2 Centering the path	32

2.3.3	Description of the method	33
2.3.4	Comparison with other work	36
2.4	Introducing the angle as a dimension	37
2.4.1	Principles	37
2.4.2	Algorithmic tricks	38
2.4.3	Results	38
2.4.4	Perspectives	39
2.5	Introducing recursivity in the Eikonal equation	39
3	Application à l'Endoscopie Virtuelle et à Plusieurs Problèmes en Imagerie Médicale	41
3.1	<i>Virtual Endoscopy</i>	42
3.1.1	Medical relevance	43
3.1.2	Problem with the path construction	45
3.1.3	Proposed solution for colonoscopy	46
3.1.4	Results on objects filled with air	48
3.1.5	Results on Arteries and vessels	49
3.1.6	Clinical study	53
3.1.7	Last developments and perspectives	58
3.2	Live-Wire	59
3.2.1	Existing methods	61
3.2.2	Adaptations and improvements	65
3.2.3	Dedicated potential	66
3.2.4	<i>Path-Cooling</i> improvement	69
3.2.5	<i>On-The-Fly</i> training improvement	70
3.2.6	Conclusion	75
3.2.7	Perspective	76
II	Extraction de Surfaces	79
4	Modèles Déformables pour l'Extraction de Surfaces en Imagerie Médicale	83
4.1	Classical Active Contours	84
4.1.1	Definition	84
4.1.2	Drawbacks	84
4.2	Geodesic Active Contours	85
4.3	Level-Sets Implementation of the Geodesic Active Contours	86
4.3.1	Advantages of this formulation	87
4.3.2	Different Motions of the interface	89
4.4	Region-based forces	92
4.4.1	Partitioning the image domain	92
4.4.2	Region descriptors	93
4.4.3	Boundary descriptors	94
4.4.4	Segmentation as an optimization process	94
4.5	Segmentation with <i>Fast-Marching</i> algorithm	96

4.6	Medical imaging applications of the <i>Level-Sets</i>	96
4.6.1	Cortex segmentation	97
4.6.2	Brain Vessels	98
4.7	Summary	99
5	Différentes Stratégies de Segmentation Basées sur les <i>Level-Sets</i> et le <i>Fast-Marching</i>	101
5.1	Interactive Segmentation with the Level-Sets Framework	102
5.1.1	Interactivity requirements for the level-sets paradigm	102
5.1.2	Fixing a point of the contour	103
5.1.3	Re-initializing the contour	104
5.1.4	Conclusion on the interactivity	106
5.2	Region Based Forces	107
5.2.1	Gaussian descriptors	107
5.2.2	Modified Gaussian descriptors	107
5.2.3	Sigmoid descriptors	108
5.2.4	Numerical implementation of the level-sets paradigm	108
5.3	Using the Fast-Marching for segmentation	112
5.3.1	Comparison with the watershed algorithm	112
5.3.2	Interactive segmentation with the Fast-Marching	114
5.4	Combining Fast-Marching and Level Sets	118
5.4.1	Advantages and Drawbacks of <i>Fast-Marching</i>	118
5.4.2	Advantages and Drawbacks of <i>Level-Sets</i>	119
5.4.3	Combining two complementary methods	120
6	Applications de la Combinaison des Techniques de <i>Fast-Marching</i> et de <i>Level-Sets</i> à l'Extraction de Surface	121
6.1	Segmentation of cerebral aneurysms	122
6.1.1	Description of the acquisition system	124
6.1.2	Application to the segmentation of brain aneurysms	124
6.1.3	Perspective	128
6.2	Detection of Colon Polyps	129
6.2.1	Segmentation of the colon surface	129
6.2.2	Visualization of the colon polyps	131
6.2.3	Perspectives	133
III	Visualisation et Quantification de Structures Anatomiques Arborescentes	137
7	Outils de Visualisation et de Mesure	141
7.1	Visualization of 3D segmentation	142
7.1.1	Virtual reality notions	142
7.1.2	Visualization of a level-set	143
7.1.3	Problem with the <i>Marching-Cubes</i>	144
7.1.4	Restoring the distance function	145

7.1.5	Volume versus Surface Rendering	147
7.2	Measurement Tools	148
7.2.1	Gauss Theorem	148
7.2.2	Volume Measurement	149
7.2.3	Example of volume measurement: an aneurysm	150
7.2.4	Section Measurement	150
8	Extraction de Structures Arborescentes	155
8.1	Introduction	156
8.2	Motivation	156
8.2.1	Tree extraction	156
8.3	Design of an adequate initialization algorithm	158
8.3.1	Propagation Freezing for Thin Structures	158
8.3.2	Suitable stopping criterion	162
8.4	Extracting the skeletal information	164
8.4.1	Combining path and shape extraction	165
8.4.2	From Trajectories to Tree Extraction	169
9	Application à l'Extraction, la Visualisation, et la Quantification d'Objets Anatomiques Arborescents	175
9.1	Application to 3D Vascular Images with Multiscale Vessel Enhancement	176
9.1.1	Medical relevance	176
9.1.2	Proposed solution	178
9.1.3	Comparisons and conclusion of the tree extraction method	179
9.2	Application to the Bronchial Tree	183
9.2.1	Medical interest	183
9.2.2	State of the art in Bronchoscopy imaging	184
9.2.3	Applying our framework	186
9.2.4	Conclusion	190
9.3	Reconstruction of vessels in 2D and 3D images using <i>Perceptual Grouping</i>	193
9.3.1	Finding Contours from a Set of Connected Components	194
9.3.2	Finding a Set of Paths in a 3D Image	199
9.3.3	Conclusion	201
Conclusion		203
Bibliography		207
Publications		217
Curriculum Vitae		219

Abbreviations

1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
3DRA	3D rotational angiography
AAA	abdominal aortic aneurysm
CAF	cost assignment function
CNR	contrast-to-noise ratio
CT	computed tomography
CTA	computed tomography angiography
CRC	colorectal cancer
DSA	digital subtraction angiography
DSR	digital subtraction radiography
fMRI	functional magnetic resonance imaging
Gd-DTPA	gadopentetate dimeglumine
GWDT	gray-weighted distance transform
MDL	minimum description length
MIP	maximum intensity projection
MPR	multi planar reformatting
MRA	magnetic resonance angiography
MRI	magnetic resonance imaging
pixel	picture element
SNR	signal-to-noise ratio
TTP	Tissue Transition Projection
US	Ultrasound (Imaging)
VOI	Volume of Interest
voxel	volume element
XRA	X-ray angiography

Introduction

Contexte

Le volume d’images médicales produites dans le monde est en constante augmentation depuis plusieurs décennies. L’utilisation d’images tridimensionnelles devient de plus en plus fréquente et les volumes de données produits par les hôpitaux demandent d’importantes ressources de traitement. Si les images 2D suffisent à de nombreuses applications et sont largement utilisées pour des raisons de coût d’acquisition, l’utilisation d’images 3D tend à se banaliser.

De l’héritage des images 2D, il reste que les images 3D sont souvent produites comme une succession de coupes. Le praticien doit mentalement “empiler” les coupes pour se faire une représentation du volume des données observées. Cela conduit à une interprétation subjective des données, et la plupart des traitements manuels coupe à coupe conduisent à une perte d’information, puisqu’une dimension n’est pas prise en compte.

L’interprétation automatique d’images médicales étudiée principalement dans cette thèse est la segmentation, ou l’isolement du reste de l’image, d’une structure anatomique d’intérêt dans le but de la visualiser, et la mesurer. Ce processus de segmentation consiste à isoler les structures visibles par délimitation de leurs contours, et le nombre de méthodes proposées en traitement d’images va croissant. On peut classifier de manière très générale les outils de segmentation en deux catégories: l’approche directe consiste à appliquer des opérateurs travaillant sur l’information de l’image directement, tandis qu’une approche par modèle fait intervenir une modélisation de l’objet recherché, en introduisant une information à priori sur l’objet recherché.

En particulier, les objets tubulaires peuvent être extraits et visualisés à l’aide de techniques utilisant un à priori sur leur forme. La meilleure manière d’étudier ce genre d’objets est d’en extraire la surface, et la structure sous-jacente, autrement dit squelette, pour les parcourir, visualiser et quantifier.

En utilisant des techniques issues de l’optique géométrique et de la propagation de la lumière dans un milieu continu, nous allons étudier précisément des implémentations rapides et efficaces d’extraction des surfaces d’objets tubulaires, ainsi que de leurs primitives géométriques, les courbes qui définissent leur squelette. Nos méthodes seront spécifiquement dédiées à l’à priori que nous avons de la forme de ces objets. Ainsi nous aurons à notre disposition un ensemble de descriptions et d’outils d’interprétation qui nous permettront tout aussi bien de visiter “virtuellement” ces objets, que de créer des outils d’aide à la décision pour tout ce qui concerne l’étendue

de pathologies, et le choix de leur traitement.

Plan

Nous proposons tout d'abord dans la première partie une étude de différents problèmes liés à l'extraction de chemins dans les images médicales 2D et 3D. Le premier chapitre est une revue des différentes techniques déjà utilisées dans ce domaine, en particulier tout ce qui est lié aux chemins minimaux, que leur formulation soit discrète ou continue. Par la suite, on se focalise sur la formulation qui est dérivée de celle des contours actifs géodésiques, et d'un schéma rapide de calcul de ces chemins, basé sur une formulation implicite du problème à l'aide d'Ensembles de Niveaux. Le second chapitre contient l'essentiel des recherches poursuivies dans le domaine de l'extraction de chemins, qu'elle soit automatique ou interactive, dans le but de faciliter cette tâche, en l'automatisant au maximum, et en réduisant les coûts qu'elle génère. Le troisième chapitre est celui qui présente deux applications de toutes les techniques du chapitre précédent. Une concerne l'endoscopie virtuelle où le chemin à extraire est celui d'une caméra virtuelle qui se déplace dans des images volumiques, et la seconde est le développement d'un outil de segmentation interactive et temps-réel, basé sur les mêmes principes jusqu'à étudiés.

Suivant le même enchaînement, la deuxième partie se penche sur l'extraction de surface, en utilisant le même formalisme des Ensembles de Niveaux que dans la première partie. Le chapitre 4 fait le tour de la question sur l'application de ce formalisme en segmentation de surfaces en imagerie médicale tridimensionnelle. Le chapitre 5 contient diverses implémentations visant à améliorer quelques défauts de ces techniques qui sont notamment: l'introduction d'interactivité dans le processus de segmentation, et la diminution des temps de calculs parfois exorbitants. Le chapitre 6 fait lui la part belle aux applications en présentant deux utilisations des méthodes d'extraction de surface: d'une part pour visualiser les anévrismes cérébraux, dont l'étude de l'étendue peut mener à optimiser le processus chirurgical nécessaire à leur traitement, et d'autre part à l'extraction et la visualisation des polypes du colon, qui sont des tumeurs pour lesquelles l'efficacité du traitement approprié dépend de manière critique de leur détection à un stade peu avancé.

La troisième partie est dédiée à l'étude de structures anatomiques beaucoup plus spécifiques: les structures arborescentes, comme tout ce qui est vasculaire ou artériel, dans le but d'optimiser la visualisation des objets, mais aussi de quantifier l'étendue des pathologies qu'ils présentent. Dans le chapitre 7, on commence par détailler les outils qui vont permettre de quantifier ces pathologies, sur la base des surfaces et courbes extraites dans l'image. Le chapitre 8 fait le lien entre tous les travaux précédemment accumulés, dans le but d'extraire dans un même processus, la surface ainsi que le squelette des objets considérés, en utilisant nos outils d'extraction de chemins et de surfaces des deux premières parties. Dans ce chapitre, les outils sont spécifiquement dédiés à la topologie si particulière des objets étudiés. Cette information de surfaces et d'arbre centré à l'intérieur de nos objets tubulaires nous amène naturellement à appliquer notre méthode, dans le chapitre 9, à l'extraction de veines et d'artères dans des images médicales 3D, puis au problème plus complexes de l'extraction de l'arbre

bronchique. Nous terminons avec une autre méthode de reconstruction d'arbres vasculaires, à l'aide de méthodes dérivées du groupement perceptuel.

Contributions

Dans la première partie notre contribution sur l'extraction de chemins est tout d'abord d'un point de vue technique d'avoir amélioré l'extraction de chemins de manière significative

- en étendant au 3D la méthode de Cohen et Kimmel [34];
- en réduisant les couts de calculs de cette méthode, et l'interactivité nécessaire;
- en développant une méthode originale d'extraction de chemins centrés;

Par ailleurs, dans cette même partie, les applications développées sur la base de nos méthodes ont mené à des implémentations fiables et validées cliniquement, en ce qui concerne l'endoscopie viruelle, suggérant de nombreuses perspectives.

Dans la seconde partie, notre contribution est avant tout d'avoir

- développé un algorithme rapide de pré-segmentation basé sur les techniques de chemins minimaux et en particulier l'algorithme du *Fast-Marching* ;
- détaillé des techniques pour améliorer l'interactivité des *Level-Sets*, méthodes connues pour leur grande précision et leur gestion des changements de topologies des objets à segmenter, mais leur faible potentiel en matière d'interactivité;
- proposé une approche collaborative à deux méthodes complémentaires, telle que le *Fast-Marching* et les *Level-Sets*, de manière plus formelle que cela avait été fait auparavant dans le domaine.

De plus, ces développements se justifient dans les applications qui sont faites par la suite dans le dernier chapitre de cette partie sur la segmentation rapide et la visualisation d'objets complexes, en imagerie médicale tridimensionnelle.

Enfin dans la dernière partie, nous présentons des contributions spécifiques dans le domaine de l'extraction et la quantification de structures tubulaires et arborescentes:

- en utilisant le même formalisme de chemins minimaux, nous présentons une technique originale pour pré-segmenter rapidement les objets tubulaires;
- nous trouvons comment récupérer simultanément un ensemble de trajectoires dans ces objets arborescents;
- nous expliquons comment remonter à l'information d'arbre à partir de ces trajectoires.

En pour finir, nous détaillons une application de ces méthodes développées précédemment à plusieurs problèmes concrets en imagerie médicale: la segmentation et la reconstruction d'arbres vasculaires, en présence de sténoses ou d'anévrismes, et d'arbres bronchiques dans des images scanners des poumons.

I

Extraction de Chemins



SELLIER.

Chapter 1

Les Chemins Minimaux en Traitement d'Images

Résumé — Les chemins minimaux sont construits à partir de l'analogie avec la propagation de la lumière dans un milieu avec un certain indice de réfraction, suivant le principe de *Pierre de Fermat*. On explique tout d'abord dans la section 1.1 le lien entre ces chemins de lumière et la théorie de la réfraction, en montrant au passage l'intérêt de l'application de cette théorie au traitement d'images.

Dans la section 1.2, partant de la formulation classique des contours actifs de *Kass, Witkin, et Terzopoulos* [82], on passe à la formulation de chemin minimal, telle qu'elle a été présentée par *Cohen et Kimmel* [34].

Mettant en parallèle la formulation discrète donnée par *Dijkstra* [43] des chemins minimaux dans des graphes avec le formalisme de *Cohen et Kimmel* [34], on explicite dans la section 1.3 différentes méthodes d'extraction de chemins, ainsi que la définition des différents termes intervenant dans ce modèle: la force liée à l'image, ou *Potentiel*.

Dans la section 1.4, on étudie le rôle du terme qui permet de contrôler la longueur du chemin, et son influence sur la courbure de ce dernier.

Abstract — Minimal paths are built upon analogy with the theory of wave-light propagation in a medium with a refractive index, according to the principles of *Pierre de Fermat*. We first explain in section 1.1 the link between this minimal light paths and the refraction principle, emphasizing the interest for the use of minimal paths in image processing.

Starting from the classical formulation of the active contours called *snakes* of *Kass, Witkin, and Terzopoulos* [82], we extend in section 1.2 to the formulation of the minimal path, as presented by *Cohen and Kimmel* [34].

Comparing the discrete version of the minimal paths given by *Dijkstra* [43] with the continuous equivalent formalism of *Cohen and Kimmel* [34], we detail in section 1.3 several implementations of extraction techniques, and the settings of parameters involved in the model, such as the image force, named *Potential*.

In section 1.4 we study the influence of the offset term which controls the length of the minimal path (and its curvature).

1.1 Minimal Paths theory

1.1.1 The minimal path in geometrical optic

In order to understand the underlying law of refraction, behind the minimal path principle, let us imagine a straight seashore, separating the sea from the beach, as shown in figure 1.1. A lifeguard sitting at a point A in the beach sees a girl drowning

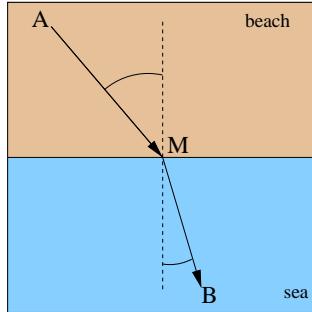


Figure 1.1. Example of minimal path in a heterogeneous media

at a point B in the sea. Assuming that the lifeguard can run on the beach three times faster than he can swim, the shortest time path is the broken line going straight from A to a point M on the shore, then straight from M to B. Considering the two angles of the straight lines to the normal to the shore, the ratio of their sines is equal to the ratio of the corresponding speeds (*Descartes/Snell's law of refraction*). The principle of *Fermat* is that light waves of a given frequency travels the path between two points which takes the least time. The most obvious example of this is the passage of a light through a homogeneous medium, in which the speed of light does not change with position. In this case, shortest time is equivalent to the shortest distance between the points, which is a straight line, as shown in figure 1.2-left. When the medium is not homogeneous, as in figure 1.2-middle, there is a refraction angle at the interface between the two homogeneous regions. Figure 1.2-right can illustrate a well-known optical phenomenon, called *mirage*: The light source S is visible from both points R_1 , and R_2 . But the light path between S and R_2 is not a straight line, due to the difference of index of the two media. Therefore, R_2 “sees” S coming from location M, at the interface between the two media, while the image source is far from M. This phenomenon occurs when the variations in temperature are important enough to deviate the light path, resulting in “visions” of an oasis in the desert, for example. Hamilton defined optical path functions, which best known was defined by Burns as the *Eikonal equation* applied to the development of a mathematical theory of optical systems. The *Eikonal equation* is used to compute the minimal light paths for a refractive index, in the sense that the minimal path is the one which integral over the refractive index is minimal.

We are going to use this minimality property in order to extract curves in images, giving only the two extremities of the path, and using the equation developed by *Hamilton* for optical systems.

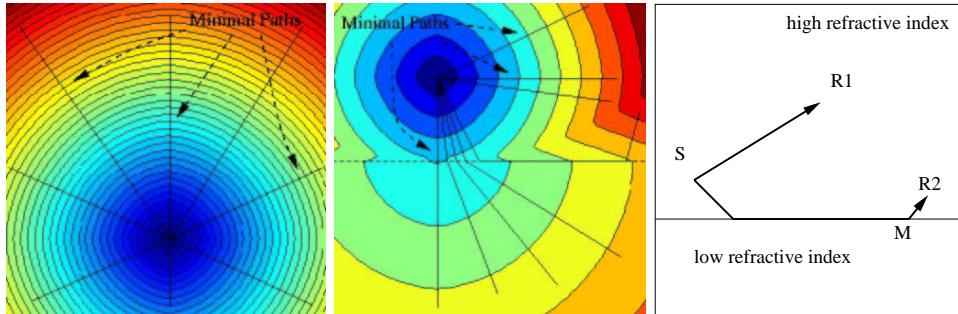


Figure 1.2. Example of minimal paths in synthetic media: left image represents the propagation of a light wave in a homogeneous medium, starting from a unique light source. The minimal paths between this light source and other position in the plane are straight lines. Middle image represents the propagation of a light wave in a medium where the refractive index is more important in the upper half part of the image than in the lower, starting from a light source in the upper part. Right image is a diagram which illustrates the *mirage* phenomenon on the basis of the propagation of the light in heterogeneous media, as shown in middle image.

1.1.2 Minimal path for curve extraction

We explain how this minimal path principle can be used for curve extraction in images. Given a refractive index P , called *potential* in the following, which takes lower values near the edges or features, our goal is to find a single contour that best fits the boundary of a given object or a line of interest. This contour, considered between two fixed extremities, will be the one which integral over the potential P is minimal.

Looking for a path which lies in a desired region of interest, the refractive index should model the desired properties of the targeted curve. For example, in figure 1.3, we want to extract a path which stays inside the vessel. The dataset, a digital sub-



Figure 1.3. Example of a minimal path in a media defined by a grey level image: The minimal path (in white) superimposed on the date is the one that corresponds to the light wave propagation, using the grey-level information as a refractive index.

tracted angiography (DSA) shows vessels in lower grey levels on a bright background. Since we want to extract a path which stays inside the vessel, a possible refractive index to be used with the *Hamilton* equations for extracting minimal paths, could be the simple image grey level values.

This ‘best fit’ question leads to algorithms that seek for the minimal path, i.e. paths along which the integration over P is minimal. Classical path extraction techniques are based on the *snakes* [82]. Snakes are a special case of deformable models as presented in [174]. Snakes start from a path close to the solution and converge to a local minimum of the energy. In this minimal path formulation, one interesting aspect is that the user input is limited to the end points, simplifying the initialization process. Unicity of this minimal path, for a given media avoids erroneous local minima. Motivated by the ideas put forward in [86, 87] *Cohen and Kimmel* developed an efficient and consistent method to find the path of minimal cost between two points, using the surface of minimal action [151, 87, 178] and the fact that operating on a given potential (cost) function helps in finding the solution for our path of minimal action (also known as minimal geodesic, or path of minimal potential). In the following we show how the formulation of the minimal light path can be obtained through a modification of the classical formulation of the active contours, and we show the numerical implementation of the minimal path extraction.

1.2 The Cohen-Kimmel Method in 2D

Starting from the famous *Snakes* model, we show how can we derive a formulation of a minimal path extraction which leads to the expression of one of the optical equations of *Hamilton*, the *Eikonal*.

1.2.1 Global minimum for Active Contours

We present in this section the basic ideas of the method introduced by *Cohen and Kimmel* [34] to find the global minimum of the active contour energy using minimal paths. The energy to minimize is similar to classical deformable models (see [82]) where it combines smoothing terms and image features attraction term (Potential P):

$$E(C) = \int_{\Omega} \left\{ w_1 \|C'(s)\|^2 + w_2 \|C''(s)\|^2 + P(C(s)) \right\} ds \quad (1.1)$$

where $C(s)$ represents a curve drawn on a 2D image, $\Omega = [0, L]$ is its domain of definition, and L is the length of the curve. Thus the curve is under the control of two kinds of forces:

- The internal forces (the first two terms) which impose the regularity on the curve. The choice of constants w_1 and w_2 determines the elasticity and rigidity of the curve.

- The image force (the potential term) pushes the curve to the significant lines which correspond to the desired attributes, for example

$$P(C) = g(\|\nabla I(C)\|). \quad (1.2)$$

Here, I denotes the image and $g(\cdot)$ is a decreasing function. In the classical snakes [82], we have $g(x) = -x^2$. The curve is then attracted by the local minima of the potential, i.e. edges (see [61] for a more complete discussion of the relationship between minimizing the energy and locating contours). Other forces can be added to impose constraints defined by the user. As introduced in [29], previous local edge detection [20] might be taken into account as data for defining the potential.

The approach introduced in [34] modifies this energy in order to reduce the user initialization to setting the two end points of the contour C . They introduced a model which improves energy minimization because the problem is transformed in a way to find the global minimum. It avoids the solution being stucked in local minima. Let us explain each step of this method.

1.2.2 Problem formulation

Most of the classical deformable contours have no constraint on the parameterization s , thus allowing different parameterization of the contour C to lead to different results. In [34], contrary to the classical snake model (but similarly to geodesic active contours), s represents the arc-length parameter, which means that $\|C'(s)\| = 1$, leading to a new energy form. Considering a simplified energy model without any second derivative term leads to the expression $E(C) = \int \{w\|C'\|^2 + P(C)\}ds$. Assuming that $\|C'(s)\| = 1$ leads to the formulation

$$E(C) = \int_{\Omega} \{w + P(C(s))\}ds \quad (1.3)$$

We now have an expression in which the internal forces are included in the external potential. In [34], the authors have related this problem with the recently introduced paradigm of the level-set formulation. In particular, its Euler equation is equivalent to the geodesic active contours [23]. The regularization of this model is now achieved by the constant $w > 0$. This term integrates as $\int_{\Omega} wds = w \times \text{length}(C)$ and allows us to control the smoothness of the contour (see [34] for details). We remove the second order derivatives from the snake term, leading to a potential which only depends on the external forces, and on a regularization term w . It makes thus the problem easier to solve, and it is used in minimal paths [34], active contours using level sets [113] and geodesic active contours as well [23]. In [34] is also mentioned how the curvature of the minimal path is now controlled by the weight term w . This corresponds to a first order regularization term, and the paths show sometimes angles. A second order regularization term would give nicer paths, but it is difficult to include such a term in the approach.

Given a potential $P > 0$ that takes lower values near desired features, we are looking for paths along which the integral of $\tilde{P} = P + w$ is minimal. The surface of minimal

action U is defined as the minimal energy integrated along a path between a starting point p_0 and any point p :

$$U(p) = \inf_{\mathcal{A}_{p_0,p}} E(C) = \inf_{\mathcal{A}_{p_0,p}} \left\{ \int_{\Omega} \tilde{P}(C(s)) ds \right\} \quad (1.4)$$

where $\mathcal{A}_{p_0,p}$ is the set of all paths between p_0 and p . The minimal path between p_0 and any point p_1 in the image can be easily deduced from this action map. Assuming that potential P is always positive, the action map will have only one local minimum which is the starting point p_0 , and the minimal path can be found by a simple back-propagation on the energy map. Thus, contour initialization is reduced to the selection of the two extremities of the path.

It is possible to compute the surface U in several ways. We are going to describe one of them that is consistent with the continuous case while implemented on a rectangular grid. It is, however, possible to implement a simple approximation like the shading from shape algorithm introduced in [178], or even graph search based algorithms ([43, 151]), if consistency with the continuous case is not important.

1.3 Numerical Implementation

The numerical schemes we propose are consistent with the continuous propagation rule. The consistency condition guarantees that the solution converges to the true one as the grid is refined. This is known **not** to be the case in general graph search algorithms that suffer from *digitization bias* due to the *metrication error* when implemented on a grid [119, 93]. This gives a clear advantage to our method over minimal path estimation using graph search. Before we introduce the proposed method, let us review the graph search based methods that try to minimize the energy given in (1.3).

1.3.1 Graph Search Algorithms and Metrication Error

Based on the new energy definition (1.3), we are able to compute the final path without evolving an initial contour, by using the surface of minimal action. To find the surface of minimal action, graph search and dynamic programming techniques were often used, where the image pixels serve as vertices in a graph [122, 53, 26], considering the image as an oriented graph.

Oriented graph

A digital image is an array of pixels. With the optimal path approach, the pixel-array is considered to be a directed graph. A local cost is associated with each node of the graph, and a link cost is associated with every arc. The costs (both local and link) are determined by the energy to minimize, therefore also called cost-function. The problem of finding the optimal boundary segment between two image pixels is transformed into finding the optimal path between two nodes in the graph. A path between two points is said optimal if the sum of the cost function values at

each of its points (called cumulative cost) is lower than the cumulative costs of any other path between the same two points. Number of algorithms, based on dynamic programming, are available in the graph theory literature. Dijkstra's algorithm [43], which computes the optimal path between a single source point to any other point in the graph, is the basis of both the *Live-Wire* and the *Intelligent Scissors*'s graph search algorithms [49, 127].

Dijkstra's path search algorithm

Path extraction algorithms like *Live-Wire* [49] and *Intelligent Scissors* [127] tools use a search method based on Dijkstra's algorithm [43]. A description of Dijkstra's algorithm, applied to road detection, can be found in [53].

The algorithm (see table 1.1) is initialized by selecting a start point s (also called seed point) in the graph. Giving this start point s , the cumulative cost of a pixel p is the sum of the cost function values of the points of the optimal path from s to p . The size of the active list is the total range of possible cumulative cost values, and its i -st item contains the list of pixels with their cumulative cost valuing i . The cumulative cost is initialized with value ∞ everywhere except at a start point s with value zero. The active list is initialized by inserting the starting pixel into the first sub-list.

At each iteration of the algorithm, the pixel p with the lowest cumulative cost is removed from the active list and expanded: the cumulative cost of each of its non-expanded neighbors is updated, and the active list is reorganized. The updating of the cumulative cost of the neighbors runs as follows: the newly computed cumulative cost of a neighbor q is the sum of the cumulative cost of its father p and the link cost from p to q . If this newly computed cost is lower than the previous one the cumulative cost of q becomes the lately calculated cost, the active list is updated and a path map structure keeps a pointer from q back to p . Since at each iteration one pixel gets a final value, and a search for the minimal vertex to update is performed, the algorithm complexity is $O(N \log_2 N)$ where N is the number of pixels in the image. In addition, a marker registers all the expanded pixels. The optimal path between the start point and another point in the image is obtained by following the pointers of the path map from the end point back to the seed point.

Our approach solves a continuous version of the problem. Sethian Fast Marching Method [161], described in section 1.3.2, has a similar complexity, yet it is consistent!

1.3.2 Fast Marching Resolution

In order to compute this map U , a front-propagation equation related to equation (1.4) is solved :

$$\frac{\partial C}{\partial t} = \frac{1}{\tilde{P}} \vec{n} \quad (1.5)$$

It evolves a front starting from an infinitesimal circle shape around p_0 until each point inside the image domain is assigned a value for U . The value of $U(p)$ is the time t at which the front passes over the point p .

Algorithm for optimal path search

- Input:
 - cost function \mathcal{P} ($\mathcal{P}(p, q)$ = cost of the arc (p, q) in the graph);
 - starting seed pixel s ;
- Output: the path map \mathcal{M} ;
- Data structures:
 - cumulative cost array CC
 - Sorted active list \mathcal{L}
 - Boolean expanded pixels marker \mathcal{E}
- Begin:
 1. put $CC(s) = 0$ and $CC(n) = \infty \forall n$ in the set of graph vertices;
 2. put s in the list \mathcal{L} ;
 3. while \mathcal{L} is not empty do
 - (a) consider p the pixel in \mathcal{L} with $CC(p)$ minimal;
 - (b) consider $\mathcal{E}p = \text{TRUE}$;
 - (c) remove p from \mathcal{L} ;
 - (d) For each neighbor q of p such that $\mathcal{E}p = \text{FALSE}$, do
 - i. $u = CC(p) + \mathcal{P}(p, q)$;
 - ii. if $u < CC(q)$ then
 - $CC(q) = u$;
 - update position of q in \mathcal{L} ;
 - put a marker from q to p in \mathcal{M} .

Table 1.1. Dijkstra Optimal Path Search Algorithm as used in [49] and [127]

The *Fast Marching* technique, introduced in [2], [161], and detailed in [163], was used by [33], noticing that the map U satisfies the Eikonal equation:

$$\|\nabla U\| = \tilde{P} \quad (1.6)$$

originally developed by *Hamilton and Burns* for geometrical optics (see section 1.1). Classic finite difference schemes for this equation tend to overshoot and are unstable. [163] has proposed a method which relies on a one-sided derivative that looks in the up-wind direction of the moving front, and thereby avoids the over-shooting associated with finite differences :

$$\begin{aligned} (\max\{u - U_{i-1,j}, u - U_{i+1,j}, 0\})^2 &+ \\ (\max\{u - U_{i,j-1}, u - U_{i,j+1}, 0\})^2 &= \tilde{P}_{i,j}^2 \end{aligned} \quad (1.7)$$

giving the correct viscosity-solution u for $U_{i,j}$. Authors of [150] also presented a direct numerical approach to solve (1.6) and gave a convergence proof to that minimization procedure in the viscosity solutions framework [35]. The principle of the *Fast Marching* is to introduce order in the selection of the grid points. This order is based on the

fact that information is propagating *outward*, because action can only grow due to the quadratic equation (1.7). Therefore the solution of equation (1.7) depends only on neighbors which have smaller values than u .

The algorithm is detailed in 3D in next section in table 2.1. The *Fast Marching* technique selects at each iteration the *Trial* point with minimum action value. This technique of considering at each step only the necessary set of grid points was originally introduced for the construction of minimum length paths in a graph between two given nodes in [43].

Thus it needs only one pass over the image. To perform efficiently these operations in minimum time, the *Trial* points are stored in a min-heap data-structure (see [161, 160, 1, 89, 163] for further details on the above algorithm, as well as a proof of correct construction). Since the complexity of the operation of changing the value of one element of the heap is bounded by a worst-case bottom-to-top proceeding of the tree in $O(\log_2 N)$, the total work is about $O(N \log_2 N)$ for the *Fast Marching* on a N points grid.

Finding the shortest path between any point p and the starting point p_0 is then simply done by back-propagation on the computed minimal action map. It consists in gradient descent on U starting from p until p_0 is reached, p_0 being its global minimum, since the geodesics are orthogonal to the wave fronts (see Bellman [11] for a nice proof on this orthogonality).

1.3.3 Back-propagation

In order to determine the minimal path between p_0 and p_1 , we need only to calculate U_0 and then slide back on the surface U_0 from $(p_1, U_0(p_1))$ to $(p_0, 0)$. The surface of minimal action U_0 has a convex like behavior in the sense that starting from any point $(q, U_0(q))$ on the surface, and following the gradient descent direction, we will always converge to p_0 . It means that U_0 has only one local minimum that is of course the global minimum and is reached at p_0 with value zero.

This is a consequence of the results in [11] that show that the light rays (geodesics, constant parameter curves) are orthogonal to the wave fronts (equal cost contours). The gradient of U is therefore orthogonal to the wave fronts since these are its level sets. The back propagation procedure is a simple steepest gradient descent. It is possible to make a simple implementation on a rectangular grid: given a point $q = (i, j)$, the next point in the chain connecting q to p is selected to be the grid neighbor (k, l) for which $U(k, l)$ is the minimal, and so forth.

1.3.4 Comparing Dijkstra and Eikonal

Since there is no difference in the overall complexity of both implementations, one may argue that using previously mentioned graph search algorithm like Dijkstra's [43, 155] might be sufficient. This algorithm is indeed efficient, but suffers from metrification errors. The graph based algorithms consider the image as an oriented graph in which each pixel is a node, and the 4 (or 8) connections to the neighboring pixels are the vertices of the graph. The different metrics used lead to very different results in a homogeneous media (see figure 1.4). Because in the L_1 metric considered for Dijkstra,

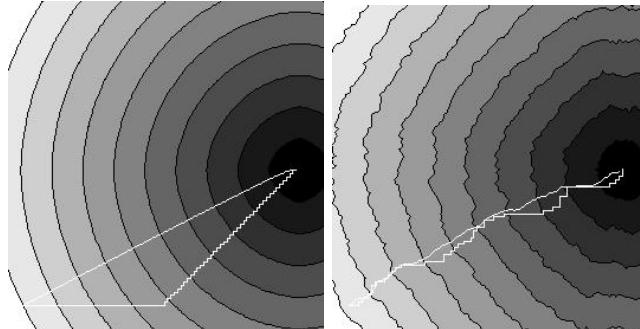


Figure 1.4. Comparing Dijksta and Eikonal optimal path extraction in a homogeneous media Left image represents both paths with the iso-action levels. Right image is the same with an added Gaussian noise.

we are limited by the distance measure imposed by the graph, how fine the grid gets. With the ‘right’ Euclidean distance measure (L^2) we get the diagonal connection as the optimal path in this case. However, notice that the homogeneous media is a synthetic dataset, and that in a noisy image, both discrete and continuous formulation lead to similar results (see figure 1.4-right).

Of course the result of the graph-search could be improved by taking a larger neighborhood as structuring element [16, 176], but there will always be an error in some direction that will be invariant to the grid resolution, which is not the case in the discretization of the *Eikonal* equation.

This error is still too important for reasons of accuracy in medical applications for example. As an illustration, figure 1.5 shows the result of searching for the minimal path in a test image.

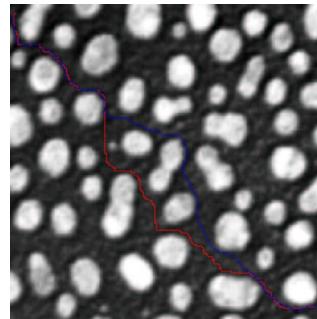


Figure 1.5. Difference between Dijksta and Eikonal on a real image: The difference is obvious in this real image example. The potential used for computing the action is the grey level information. Using the same extremities, the red path corresponds to Dijksta’s, and the blue one to Eikonal.

1.4 The Regularity of the Path

In [34], it is proven that weight w in equation (1.3) can influence curvature and be used as a smoothing term. An upper bound for the curvature magnitude $|\kappa|$ along the minimal path is found, \mathcal{I} being the image domain:

$$|\kappa| \leq \frac{\sup_{\mathcal{I}} \|\nabla P\|}{w} \quad (1.8)$$

1.4.1 Influence on the gradient descent scheme

The exact minimal path is obtained with a gradient descent. But care must be paid on the choice of the gradient step to avoid oscillations.

If the weight w is set to a small value ϵ the extracted path length is not limited at all, nor the curvature magnitude in equation (1.8). Therefore in zones where the action map is flattened, the slope being as small as ϵ , the path can have a spaghetti-like trajectory. The minimal path being obtained by steepest gradient descent, directions are evaluated by interpolation based on nearest neighbors on the Cartesian grid. If the discrete gradient step Δx is too large, the approximation of this trajectory will produce oscillations between relative positions. Those oscillations can lead to a huge number of path points larger than forecasted allocations.

We have made a test on a region of the data shown in figure 1.6-left where the steepest gradient fails (with a number of path points limited). The cost map when

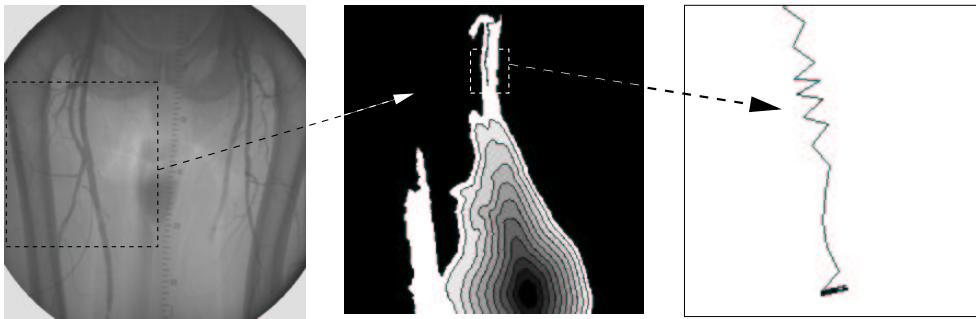


Figure 1.6. Failure of the steepest gradient descent on a bolus chase reconstruction data

tracking a vessel is displayed in figure 1.6-middle. Taking $w = .1$ leads to a curvature magnitude $\kappa \leq 10^3$. The steepest gradient scheme oscillates, for a given step size, and stops as shown in figure 1.6-right. Therefore, increasing w maintains a lower upper-bound on the curvature magnitude and makes the steepest gradient descent scheme robust. Another method is to use more robust gradient descent techniques like Runge-Kutta where the step size of the gradient descent can be locally adapted.

1.4.2 Influence on the number of points visited

This section illustrates the influence of the weight w of equation (1.3) on the necessary number of voxels visited for a path extraction. In figures 1.7 is shown the tracking of a vessel in a X-Ray image of the femoral vessels, using different weights $w_1 = 1$ and $w_2 = 20$. The smoothing done by increasing the weight can be observed in a

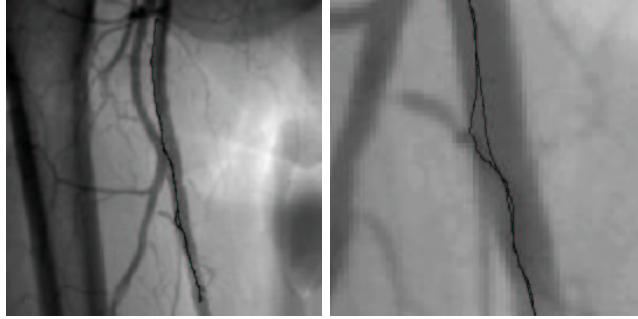


Figure 1.7. Smoothing the minimal path with the weight w : Left image shows the dataset of femoral vessels with two paths superimposed. Right image shows a zoom on the paths with $w_1 = 1$ and $w_2 = 20$.

zoom on the paths shown in figure 1.7-right. We can also observe the influence of increasing the weight in figure 1.8 where each path is displayed superimposed on its respective action map. For a small weight $w_1 = 1$, the path is not smoothed, as shown

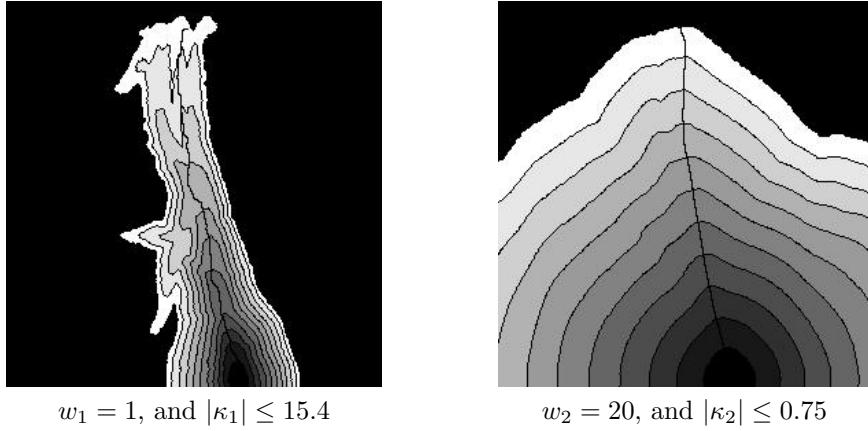


Figure 1.8. Smoothing the minimal path with the weight w : the action maps

in figure 1.8-left. For a weight $w_2 = 20$, leading to the inequality $|\kappa_2| \leq 0.75$, the path is smooth. Differences appear also in the sets of points visited during propagations: it is smaller with weight $w_1 = 1$. It means that propagation is quicker for small weights. It propagates in every directions for a higher weight (see figure 1.8-right), because the

tune of w smoothes the image, as it reduces the upper-bound on curvature magnitude in equation (1.8).

Chapter 2

Nouvelles Techniques d'Extraction de Chemins à l'Aide du *Fast-Marching*

Résumé — Ce chapitre contient diverses améliorations de la méthode originale du chapitre 1, valables aussi bien en 2D qu'en 3D.

Nous commençons par présenter en section 2.1 une extension au 3D de la méthode classique. Nous détaillons en section 2.2 des méthodes pour accélérer l'extraction de chemins et la rendre plus facile, en réduisant les interactions nécessaires. Nous développons une méthode pour extraire des chemins centrés dans des structures tubulaires en section 2.3. En section 2.4 nous calculons des trajectoires pour des objets en mouvement, en introduisant l'angle comme dimension supplémentaire au problème. Finalement, nous expliquons l'introduction d'un facteur de récursivité dans le *Fast-Marching*, afin d'extraire des chemins plus longs. Chaque technique est illustrée par un exemple sur une image réelle ou de synthèse.

Abstract — This chapter will detail various improvements and modification of the original method of chapter 1 that are useful for image analysis in 2D as well as in 3D.

In section 2.1, we extend the method detailed in section 1.3 for 2D images to 3D. In section 2.2 we give details about our techniques to make the path extraction scheme faster and easier, by reducing the user interaction. In section 2.3 we develop a new method to extract a path centered in a tubular structure. In section 2.4 we compute trajectories for moving objects, introducing a degree of freedom on their angulation. Finally, in section 2.5, we explain the introduction of a recursivity factor in the *Fast-Marching* algorithm, which enables to extract longer paths. Synthetic and real medical images are used to illustrate each contribution.

2.1 3D Extension

We are interested in this section in finding a minimal curve in a 3D image. One application that can motivate this problem is detailed in section 3.1. It can also have many other applications. Our approach is to extend the minimal path method of previous section to finding a path $C(s)$ in a 3D image minimizing the energy:

$$\int_{\Omega} \tilde{P}(C(s))ds \quad (2.1)$$

where $\Omega = [0, L]$, L being the length of the curve. An important advantage of level-set methods is to naturally extend to 3D. We first extend the *Fast Marching* method to 3D to compute the minimal action U . We then introduce different improvements for finding the path of minimal action between two points in 3D. In the examples that illustrate the approach, we see various ways of defining the potential P .

Similarly to previous section, the minimal action U is defined as

$$U(p) = \inf_{\mathcal{A}_{p_0,p}} \left\{ \int_{\Omega} \tilde{P}(C(s))ds \right\} \quad (2.2)$$

where $\mathcal{A}_{p_0,p}$ is now the set of all 3D paths between p_0 and p . Given a start point p_0 , in order to compute U we start from an initial infinitesimal front around p_0 . The 2D scheme equation (1.7) is extended to 3D, leading to the scheme

$$\begin{aligned} (\max\{u - U_{i-1,j,k}, u - U_{i+1,j,k}, 0\})^2 &+ \\ (\max\{u - U_{i,j-1,k}, u - U_{i,j+1,k}, 0\})^2 &+ \\ (\max\{u - U_{i,j,k-1}, u - U_{i,j,k+1}, 0\})^2 &= \tilde{P}_{i,j,k}^2 \end{aligned} \quad (2.3)$$

giving the correct viscosity-solution u for $U_{i,j,k}$. The algorithm which gives the order of selection of the points in the image is detailed in table 2.1. It should be noted that a generalization of this algorithm was recently introduced in [92].

Considering the neighbors of grid point (i, j, k) in 6-connexity, we study the solution of the equation (2.3). We note $\{A_1, A_2\}$, $\{B_1, B_2\}$ and $\{C_1, C_2\}$ the three couples of opposite neighbors such that we get the ordering $U_{A_1} \leq U_{A_2}$, $U_{B_1} \leq U_{B_2}$, $U_{C_1} \leq U_{C_2}$, and $U_{A_1} \leq U_{B_1} \leq U_{C_1}$. To solve the equation, three different cases are to be examined sequentially in table 2.2. We thus extend the *Fast Marching* method, introduced in by *Adalsteinsson and Sethian* [2], and used by *Cohen and Kimmel* [34] to our 3D problem.

2.2 Several minimal path extraction techniques

In this section, different minimal path extraction procedures are detailed. We present new back-propagation techniques for speeding up extraction, a one end-point path extraction method to reduce the need for interaction, and in next section, a centering path extraction method adapted to the problem of tubular structures in images. The methods presented in this section are valid in 2D as well as in 3D and this is an

Algorithm for 3D Fast Marching

- Definition:
 - *Alive* is the set of all grid points at which the action value has been reached and will not be changed;
 - *Trial* is the set of next grid points (6-connectivity neighbors) to be examined and for which an estimate of U has been computed using equation 2.3;
 - *Far* is the set of all other grid points, for which there is not yet an estimate for U ;
- Initialization:
 - *Alive* set is confined to the starting point p_0 , with $U(p_0) = 0$;
 - *Trial* is confined to the six neighbors p of p_0 with initial value $U(p) = \tilde{P}(p)$;
 - *Far* is the set of all other grid points p with $U(p) = \infty$;
- Loop:
 - Let $(i_{min}, j_{min}, k_{min})$ be the *Trial* point with the smallest action U ;
 - Move it from the *Trial* to the *Alive* set (i.e. $U_{i_{min}, j_{min}, k_{min}}$ is frozen);
 - For each neighbor (i, j, k) (6-connectivity in 3D) of $(i_{min}, j_{min}, k_{min})$:
 - * If (i, j, k) is *Far*, add it to the *Trial* set and compute U using table 2.2;
 - * If (i, j, k) is *Trial*, recompute the action $U_{i, j, k}$, and update it.

Table 2.1. Fast Marching algorithm

important contribution that can be useful for image analysis in general, for example in radar applications [8], in road detection [117], or in finding shortest paths on surfaces [86].

Examples in 2D are used to make the following ideas easier to understand. We also illustrate the ideas of this section on two synthetic examples of 3D front propagation in figures 2.1 and 2.2. Examples of minimal paths in 3D real images are presented in chapter 3.

The minimal action map U computed according to the discretization scheme of equation (2.2) is similar to convex, in the sense that its only local minimum is the global minimum found at the front propagation start point p_0 where $U(p_0) = 0$. The gradient of U is orthogonal to the propagating fronts since these are its level sets. Therefore, the minimal action path between any point p and the start point p_0 is found by sliding back the map U until it converges to p_0 . It can be done with a simple steepest gradient descent, with a predefined descent step, on the minimal action map U , choosing

$$p_{n+1} = p_n - \text{step} \times \nabla U(p_n). \quad (2.6)$$

More precise gradient descent methods like Runge-Kutta midpoint algorithm or Heun's

Algorithm for 3D Up-Wind Scheme

1. Considering that we have $u \geq U_{C_1} \geq U_{B_1} \geq U_{A_1}$, the equation derived is

$$(u - U_{A_1})^2 + (u - U_{B_1})^2 + (u - U_{C_1})^2 = \tilde{P}^2 \quad (2.4)$$

Computing the discriminant Δ_1 of equation (2.4) we have two possibilities

- If $\Delta_1 \geq 0$, u should be the largest solution of equation (2.4);
 - If the hypothesis $u > U_{C_1}$ is wrong, go to 2;
 - If this value is larger than U_{C_1} , go to 4;
- If $\Delta_1 < 0$, at least one of the neighbors A_1 , B_1 or C_1 , has an action too large to influence the solution. It means that the hypothesis $u > U_{C_1}$ is false. Go to 2;

2. Considering that we have $u \geq U_{B_1} \geq U_{A_1}$ and $u < U_{C_1}$, the new equation derived is

$$(u - U_{A_1})^2 + (u - U_{B_1})^2 = P^2 \quad (2.5)$$

Computing the discriminant Δ_2 of equation (2.5) we have two possibilities

- If $\Delta_2 \geq 0$, u should be the largest solution of equation (2.5);
 - If the hypothesis $u > U_{B_1}$ is wrong, go to 3;
 - If this value is larger than U_{B_1} , go to 4;
- If $\Delta_2 < 0$, B_1 has an action too large to influence the solution. It means that $u > U_{B_1}$ is false. Go to 3;

3. Considering that we have $u < U_{B_1}$ and $u \geq U_{A_1}$, we finally have $u = U_{A_1} + P$. Go to 4;

4. Return u .

Table 2.2. Solving locally the upwind scheme

method can be used for this path extraction. A simpler descent can be choosing $p_{n+1} = \min_{\{\text{neighbors of } p_n\}} U(p)$, but it gives an approximated path in the L_1 metric. Such a descent has no more the property of being consistent. As an example, see in figure 2.1 the computed minimal action map for a 3D Homogeneous medium defined by $P(i, j, k) = 1 \forall (i, j, k)$.

Figure 2.2 shows a front propagation on a synthetic binary example, based on a spiral. We extract a path that goes from the interior of the spiral, and finds its way out of it to the second end point outside the object.

2.2.1 Partial Front Propagation

An important issue concerning the back-propagation technique is to constrain the computations to the necessary set of pixels for one path construction. Finding several paths inside an image from the same seed point is an interesting task, but in case we have two fixed extremities as input for the path construction, it is not necessary to propagate the front on all the image domain, thus saving computing time. Figure 2.3 compares the sets of pixels visited using a classical front propagation, and a partial

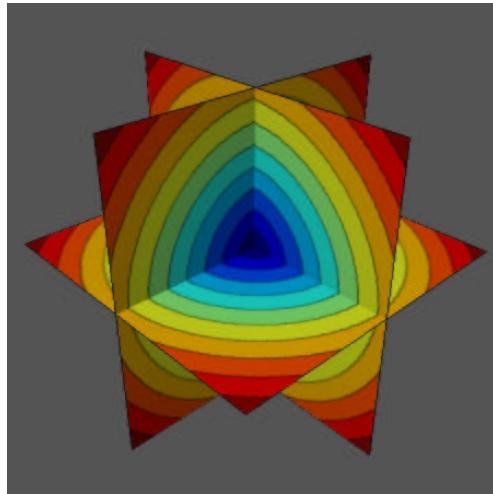


Figure 2.1. 3D visualization of an action map: these are the level sets of the action obtained by propagating a front in a homogeneous medium (constant Potential: $\forall (x, y, z) \in \mathbb{R}^3 P(x, y, z) = c > 0$) , represented with different colors. The domain is cubic, and the starting point for the wave equation is located at the center of the cube. The iso-action surfaces are concentric spheres with the starting point as center.

propagation on a Digital Subtracted angiography (DSA) image of the brain vessels. It highlights the fact that the set of points visited is smaller when propagation is only partial. We can see that there is no need to propagate further the points examined in figure 2.3-right, the path found being exactly the same as in figure 2.3-middle where front propagation is done on all the image domain. We used a potential $P(\mathbf{x}) = |\nabla G_\sigma * I(\mathbf{x})| + w$, where I is the original image (512^2 pixels, displayed in figure 2.3-left), G_σ a Gaussian filter of variance $\sigma = 2$, and $w = 1$ the weight of the model. In figure 2.3-right, the partial front propagation has visited less than half the image. This ratio depends mainly on the length of the path tracked.

2.2.2 Simultaneous Front Propagation

The idea is to propagate simultaneously a front from each end point p_0 and p_1 . Let us consider the first grid point p where those front collide. Since during propagation the action can only grow, propagation can be stopped at this step. Adjoining the two paths, respectively between p_0 and p , and p_1 and p , gives an approximation of the exact minimal action path between p_0 and p_1 . Since p is a grid point, the exact minimal path might not go through it, but in its neighborhood. Basically, there exists a real point p^* , which nearest neighbor on the Cartesian grid is p which belongs to the minimal path. Therefore, the approximation done is sub-pixel and there is no need to propagate further. This *colliding fronts* method is described in table 2.3.

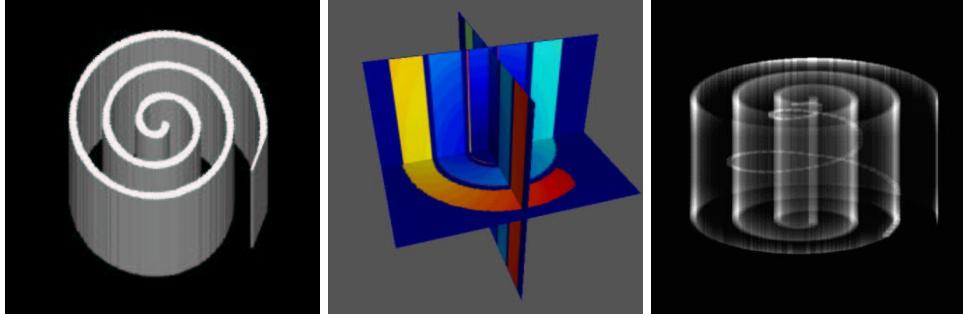


Figure 2.2. Front propagation in a synthetic 3D example: The left image is a volume rendering of the synthetic dataset, a spiral image, with a very high value ($P(x, y, z) = c_1 = 10^4$) in the spiral walls and a very low value in the background ($P(x, y, z) = c_2 = 1$). We extract the minimal path between a starting point located inside the spiral, and another one outside the spiral. The middle image represents the level sets of the action, mapped on three orthogonal planes, using the same color-map than in figure 2.1. The right image represents a transparent view of the object and the extracted path obtained.

Algorithm

- Compute the minimal action maps U_0 and U_1 to respectively p_0 and p_1 until they have an *Alive* point p_2 in common;
- Compute the minimal path between p_0 and p_2 by back-propagation on U_0 from p_2 ;
- Compute the minimal path between p_1 and p by back-propagation on U_1 from p_2 ;
- Join the two paths found.

Table 2.3. Minimal Path as intersection of two action maps

It has two interesting benefits for front propagation:

- It allows a parallel implementation of the algorithm, dedicating a processor to each propagation;
- It decreases the number of pixels examined during a partial propagation. With a potential defined by $P = 1$, the action map is the Euclidean distance.
 - In 2D (figure 2.5), this number is divided by $\frac{(2R)^2}{2 \times R^2} = 2$;
 - In 3D (figure 2.1), this number is divided by $\frac{(2R)^3}{2 \times R^3} = 4$.

Figure 2.4 compares the sets of pixels visited using a partial front propagation, as explained in section 2.2.1, and a simultaneous propagation on a Digital Subtracted angiography (DSA) image of the brain vessels. It highlights the fact that the set of

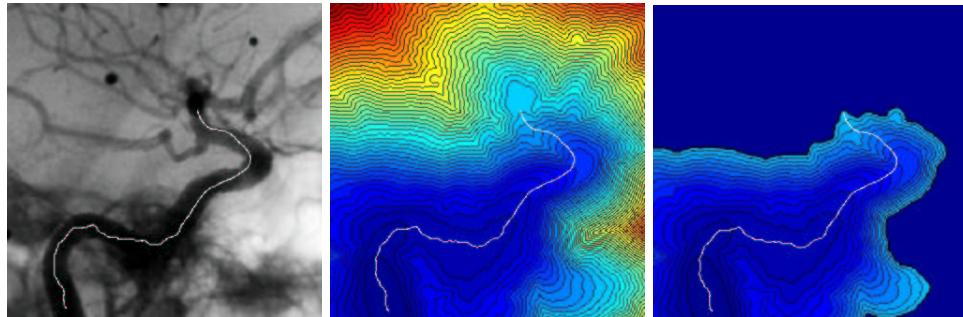


Figure 2.3. Comparing classical and partial propagation: The left image is the data set, used as potential to extract a path which stays inside a brain vessel of a DSA image; the extremities of the path are located manually. The center image is the action map obtained by propagating on the whole image domain. The right image shows the action map resulting from a partial computation. The two paths extracted are the same, due the minimality principle.

points visited is even smaller when propagation is done from both the extremities of the path.

The potential used is $P(\mathbf{x}) = |I(\mathbf{x}) - C| + w$, where I is the original image (256×256 pixels, displayed in figure 2.4-(a)), C a constant term (mean value of the start and end points gray levels), and $w = 10$ the weight of the model. In figure 2.4-(b), the partial front propagation has visited up to 60% of the image. With a colliding fronts method, only 30% of the image is visited (see figure 2.4-(c)), and the difference between both paths found is sub-pixel (see figure 2.4-(d) where the paths superimposed on the data do not differ).

The diagram in figure 2.5 represents the theoretical difference of domains visited by the algorithm, for partial and simultaneous propagations.

2.2.3 Euclidean Path Length Computation

We have shown the ability of the front propagation techniques to compute the minimal path between two fixed points. In some cases, only one point should be necessary, or the needed user interaction for setting a second point is too tedious in a 3D image. Here we derive a method that builds a path given only one end point and a maximum path length.

As we explain below, we can compute simultaneously at each point the energy U of the minimal path and its length. We choose as end point the first point for which the length of the minimal path has reached a given value. Since the front propagates faster along lower values of Potential, interesting paths are longer for a given value of U .

The technique is similar to that of section 2.2.1, but the new condition will be to stop propagation when the first path corresponding to a chosen Euclidean distance is extracted. Since the front propagates in a tubular structure, all the points for which

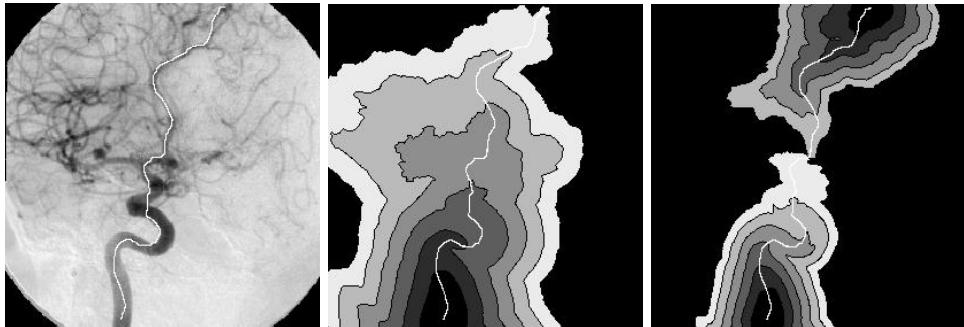


Figure 2.4. Comparing partial and simultaneous propagation: The left image is the data set, used as potential to extract a path in a vessel tree in a DSA image; the extremities of the path are located manually. The center image is the action map obtained by partial front propagating as explained in section 2.2.1. The right image shows the action map resulting from a simultaneous propagation from both extremities of the path. The second path extracted is a sub-pixel approximation of the first one, as detailed in the section.

the path length criterion is reached earlier in the process are located in the same area, far from the start point. Therefore the first point for which the length is reached is located in this area and is a valuable choice as endpoint.

Figure 2.6 represents the propagation of a front according to the potential shown in figure 2.6-left, with the *on-the-fly* computation of the approximate Euclidean length of the paths at each pixel crossed by the front. The propagation is done on the whole image domain, and one can observe that the resulting map, in figure 2.6-right is non-smoothed, and very difficult to analyze.

Figure 2.7 represents the same computation of the Euclidean path length than in figure 2.6-left, but limited to the necessary set of pixels visited in order to extract the minimal path super-imposed on the three images (the method is detailed in section 2.2.1). One can observe that the resulting map, in figure 2.6-right is non-smoothed, but we can clearly visualize the level-sets of the Euclidean path length computed at the same time.

2.3 Path centering in linear objects

The path is the set of locations that minimize the integral of the potential in equation (1.3). If the potential is constant in some areas, it will lead to the shortest Euclidean path. The same thing happens when the potential does not vary much inside a tubular shape. The minimal path extracted is often tangential to the edges, and would not be tuned for a problem which may require a centered path. Figure 2.8 describes the practical problem we are facing using the classical wave equation model of [34], in tube shaped structures where the potential is approximately homogeneous inside the object.

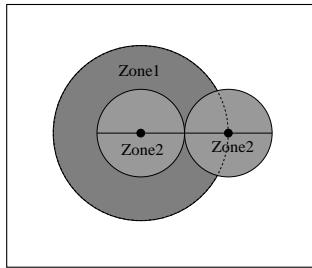


Figure 2.5. Comparing the theoretical domains for extracting a minimal path between two points in a homogeneous medium: The area labeled **Zone 1** corresponds to the needed set of pixels visited with partial propagation. The area labeled **Zone 2** corresponds to the needed set of pixels with simultaneous propagation.

The general framework for obtaining a centered path is the following

- Segmentation : the first goal is to obtain the edges of the tubular region;
- Centered path : once we have this segmented region, we want to find a path that is as much centered as possible in it. In order to attract the minimal path to the center of the region, we use a distance map from the segmented edges.

In the following we are going to present our method, introduced in [42], detailing each step and making comparisons with other existing techniques.

2.3.1 Segmentation step

In order to find the tubular structure, several approaches can be used. We can use a balloon model [29] with a classical snake approach that inflates inside the object, starting with the given end point. Or we can segment the object using its correspondent level-sets implementation, as in [113] and like the bubbles in [172]. In fact, this kind of region growing method can also be implemented using the *Fast Marching* algorithm. This fast approximation has already been used for segmentation in [111]. This allows us to include the segmentation step in the same framework as our minimal path finding: having searched for the minimal action path between two given points, using a partial front propagation (see section 2.2.1), the algorithm provides different sets of points:

- the points whose action is set and labeled *Alive*;
- the points not examined during the propagation and labeled *Far*;
- the points at the interface between *Alive* and *Far* points, whose actions are not set, and labeled *Trial*.

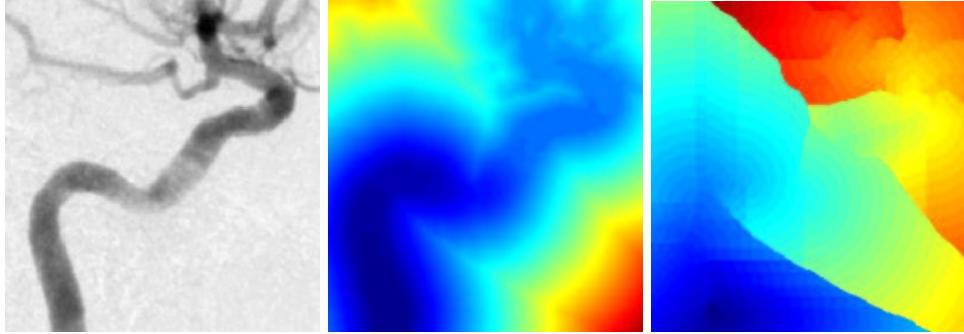


Figure 2.6. Computing the approximate Euclidean path length while propagating on the whole image: using the left image as potential, the front is propagated on the whole image domain. Middle image and right images represent respectively the action map starting from the bottom of the vessel, and the Euclidean path length computed at the same time.

This last category, the border of the visited points, is a contour in 2D and a surface in 3D which defines a connected set of pixels or voxels. If the potential is a lot higher along edges than it is inside the shape, the edges will act as an obstacle to the propagation of the front. Therefore, the front propagation can be used as a segmentation procedure, recovering the object shapes. In this case the *Trial* points define a surface which can be described as a rough segmentation. Once the front has reached the endpoint, we use the front itself to define the edges.

2.3.2 Centering the path

Having obtained this interface of *Trial* points, we now want the information of distance to the edges. This information can be either used for a skeletonization, computing the medial-axis transform, or used as a new snake energy, that constrains the path in the center of the tubular shape.

In order to compute this distance, we can use a second front propagation procedure. The edges are stored in the min-heap data-structure (see [163] for details), and this is a very fast re-initialization process to compute this distance. The potential and the initial action for this second front propagation are defined as follows:

$$\begin{aligned} P(i, j) &= 1 && \forall(i, j) \text{ inside the shape} \\ P(i, j) &= \infty && \forall(i, j) \text{ outside the shape} \\ U(i, j) &= 0 && \forall(i, j) \in \{\text{Trial}\} \text{ points of section 2.3.1} \\ U(i, j) &= \infty && \text{elsewhere} \end{aligned}$$

Starting the front propagation from all the points stored in the min-heap data-structure, we compute the distance map, said \mathcal{E} , very quickly, visiting only the pixels inside the tubular object.

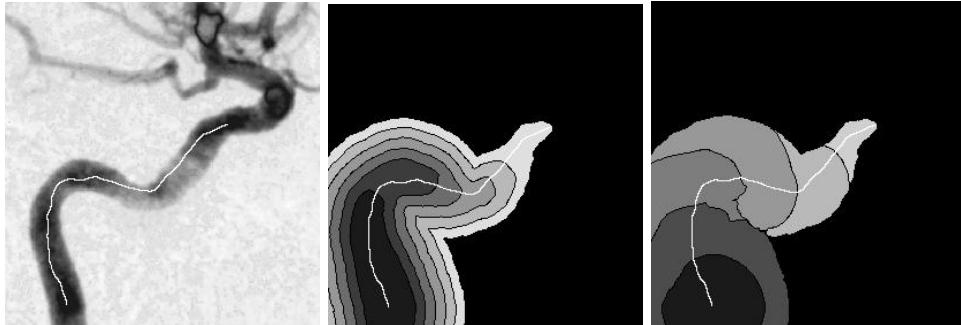


Figure 2.7. Computing the approximate Euclidean path length while propagating partially on the image domain: Left image is the potential. Middle image and right images represent respectively the action map starting from the bottom of the vessel, and the Euclidean path length computed at the same time.

Our distance map \mathcal{E} is used to create a second potential P_1 . Choosing a value d to be the minimum acceptable distance to the walls, we propose the following potential:

$$P_1(\mathbf{x}) = \max(d - \mathcal{E}(\mathbf{x}); 0)^\gamma \quad (2.7)$$

This distance d is illustrated on figure 2.9. We use this potential (2.7) for a new front propagation approach: P_1 weights the points in order to propagate faster a new front in the center of the desired regions. This final propagation produces a path centered inside the tubular structure in a very fast process.

2.3.3 Description of the method

The complete method is described in figure 2.10.

1. Segmentation: the first step is to compute the weighted distance map given the start and end points. It is obtained by front propagation from the start to the end point. Notice also that the end point can be determined automatically by a length criterion as in section 2.2.3;
2. Segmentation: the second step is to consider the set of points which have same minimal action as the endpoint. For this, we store the front position (set of trial points) at the end of the first step.
3. Centering Potential : the third step is to compute the distance map \mathcal{E} to the boundary front inside the tubular region. For this we propagate inward the front with a uniform potential $P = 1$. This gives the higher values towards the center of the object.
4. Centered path : the fourth step is to find the minimal path between start and end points relatively to the distance potential P_1 defined in (2.7) computed from the previous step. This is obtained by applying again the minimal path

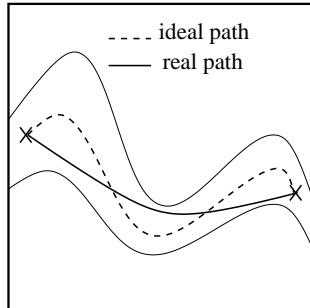


Figure 2.8. Path extraction in a tube-like object: the path obtained using the classical wave equation model is minimal according to the weighted metric, which means that it is the shortest in the tube considered; the ideal path would stay in the center of the object, as shown in the diagram.

technique. The front is now pushed to propagate faster in the center of the object.

5. Centered path : the final step is to make back-propagation from the end point using the last minimal action map.

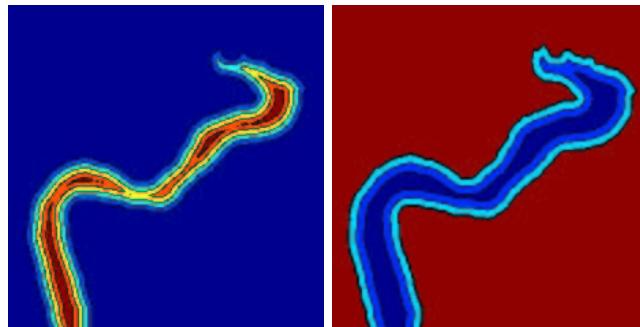


Figure 2.9. Thresholding the distance map: The left image shows the level sets of the distance to the object borders; The right image is the potential obtained by applying a threshold to the distance in order to propagate faster in a region of the tube, at a minimum distance to the borders, given as parameter.

Figure 2.10 explains the different steps of the path centering process.

An interesting improvement is that the value of the weight w can be automatically set to a very low value:

- During the first propagation the regularity of the path is not important, and w can be very small;

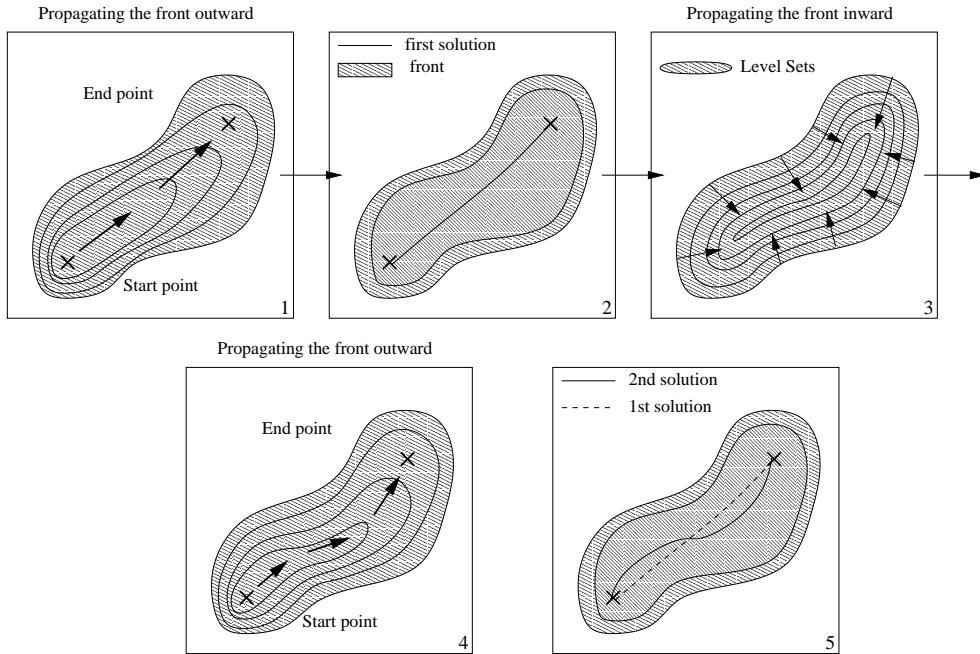


Figure 2.10. The complete path centering process: step 1 is propagation in the medium between the two extremities of the path; step 2 is to consider and store the envelope of the pixels visited during step 1; step 3 is to propagate backward into the object, computing the distance to its borders; step 4 is to use the distance as a new penalty to propagate; step 5 is to backtrack the final centered path.

- During the second propagation, $P' = P + w = 1$
- During the final propagation the potential based on the distance to the object walls is synthetic and leads to smooth paths even if $w \ll 1$.

Figure 2.11 compares the result of the classical path extraction, and the centering process detailed below, on a potential defined by a DSA image of the brain vessels. The two paths are represented super-imposed on the data in figure 2.11-left, in order to highlight the result of the modification of the penalty according to the distance to the object.

In figure 2.11-middle is shown the result obtained using a potential based on the image, where the shortest path is tangential to edges. But the front propagates only along the vessel direction, and is rapidly stopped transversally, allowing to compute the distance to the walls. Defining a new potential according to equation (2.7) based on this distance map, the second front propagates faster in the center of the vessel, at the distance d chosen. Due to the shape of the iso-action lines of the centered minimal action shown in figure 2.11-right, the path avoids the edges and remains in the center of the vessel.

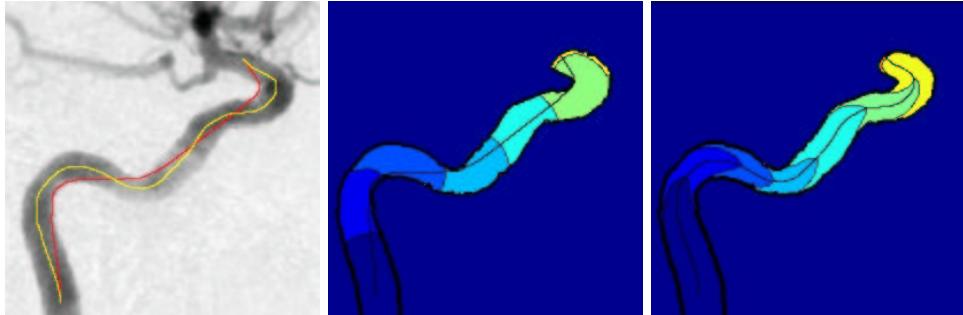


Figure 2.11. Path centering test on a 2D DSA image: Applying the process detailed in figure 2.10, the two paths extracted are super-imposed on the data on the left image; the middle image represents the level-sets of the action map with a classical use of the Eikonal equation; the right image shows the corresponding result with the centering strategy.

2.3.4 Comparison with other work

Another method to obtain a centered path would be to make a classical snake minimization on the centering potential P_1 , starting from the path obtained previously, like it is done in the thesis [36]. But too much smoothing may lead to a wrong path. For example, in the case of thin tubular structures, smoothing the path may lead it outside the tubular structure. Also, the unpublished work presented in [36] details an algorithm which is applied to a tubular object which is already manually segmented by the user, whereas our method comprises both steps of segmentation and centering.

Another category of very similar centered line extraction technique is skeletonization, and particularly the definition of the medial axis function of [15] which treats all boundary pixels as point sources of a wave front. Considering that the *Fast Marching* computes the Euclidean distance to an arbitrary set of points using a potential $P = 1$, it can also be used for skeletonization.

However, the purpose of our application is to have a smooth line which always stays inside the tubular object and which is far from the edges.

If one wishes to achieve this task with a skeletonization, like in [192], he will need and rely on the results of post-processing techniques in order to obtain a unique and smooth path inside this segmented object. Smoothing and removing undesirable small parts of the skeleton can be done using techniques shown in [173]. The main advantage of our approach is that it gives only one smooth and centered path in a unique and fast process. Therefore, it cannot be replaced by a simple medial-axis transform.

In [136], the authors extract first the surface of the colon, then compute a minimal path on this surface and move this initial path to the center of the object by applying a thinning algorithm to the object segmented and projecting the path on the resulting

surface. The algorithm developed by [90] can be applied to their methods since it computes the minimal path on a surface defined by a manifold. Although it seems to produce a smooth centered line, the thinning algorithm is computationally inefficient, compared to the speed of our algorithm that needs less than a minute on a classical inexpensive computer (300MHz CPU).

In the different techniques quoted, the main difference with our method lies in the fact that the object is manually segmented by the user. Our method comprises steps of segmentation and path extraction, and achieves them in a very fast way. More than a robust and fast method, we have developed a tool that is used for segmentation, minimal path tracking, and even potential definition. The main advantage of our approach is that it comprises all those steps and gives only one smooth and centered path in a unique and fast process.

2.4 Introducing the angle as a dimension

2.4.1 Principles

In [92, 91] the authors consider the problem of robot navigation with constraints and rotation, introducing a third degree of freedom in two-dimensional applications. Considering now an object with a given length and width, the problem is now to extract a trajectory between two positions that are in configuration space the position of the center of the object, plus an angle θ between 0 and 2π at the beginning and at the end of the trajectory.

The authors consider two cases, both on constant potentials:

- In the absence of obstacles, the *Fast-Marching* can be applied in a straightforward manner, by discretizing the configuration space into a 3D grid, namely gridding both \mathbb{R}^2 and $[0; 2\pi]$ with periodic boundaries in θ , and solving

$$[u_x^2 + u_y^2 + u_\theta^2]^{\frac{1}{2}} = 1 \quad (2.8)$$

- In the presence of obstacles, by altering the shapes of the obstacles for every discretized angle θ_i , rather than maneuvering the robot. They use morphological operations, like dilatation to alter the obstacles shapes, with a structuring element corresponding to the robot at a given angle. A fast implementation of these morphological operations can be found in [64].

Therefore the above path planning problem solves the Eikonal equation

$$|\nabla T| = \frac{1}{V(x, y, \theta)} \quad (2.9)$$

where F is binary: 1 in reachable regions and 0 inside the obstacles. As shown in [163], there is no reason to limit ourselves to binary speed values. We may use the same algorithm for continuously varying speed functions.

In [88], they consider the problem of obstacle avoidance navigating under a potential function which penalizes the free work space [100].

We worked upon the use of the Eikonal equation, including a dimension related to the angle of an object, in order to compute trajectories of oriented objects in domains with a weighted metric, without obstacles.

The problem has been schematized to the following:

1. We have used very simple objects, like rectangles and triangles, in two dimensional media;
2. For those objects, our strategy is to discretize them in a limited number of positions, which means that for a triangle, we only consider the value of the potential at its vertices.
3. In order to make the objects move according to the weighted metric, staying in the desired regions, at each position (x, y, θ) , we take as potential for an object, the maximum of the potential over all positions considered (i.e. vertices for the triangle case).

We want to simulate the trajectory of an object, in a two dimensional medium, with constraints on the direction of the object: there is now a cost for changing its orientation. This result finds its application in the regularization of the point of view of the object in the media, simulating for example the direction of a virtual camera.

2.4.2 Algorithmic tricks

The algorithm used is very similar to that of section 2.1, which means we are working on a three dimensional problem. The following definitions are necessary:

1. The speed of the front is a function of the position, but not the orientation:

$$V(x, y, \theta) = V(x, y) \quad \forall (x, y, \theta) \in \mathbb{R}^2 \times [0; 2\pi];$$
2. The action computed according to equation (2.9) is defined on $\mathbb{R}^2 \times [0; 2\pi]$, with periodic boundaries in θ .
3. the gridding of the interval $[0; 2\pi]$ used in the tests was to consider 10 discretized angle; it can be easily implemented using an array.

2.4.3 Results

Figure 2.12 represents samples of the trajectory of a triangle, using a DSA image as weighted metric for propagating. Obviously, the object is doing several U-turns along the trajectory and is not suitable for the applications we want to address.

We overcome the drawback of the very simple object used in figure 2.12-left by simply adding a branch to our triangle. This branch defines a new position for estimating the potential, and will constrain our object to look in the direction of the trajectory, in linear structures, like vessels. Results of this new strategy are shown in figure 2.12-right.

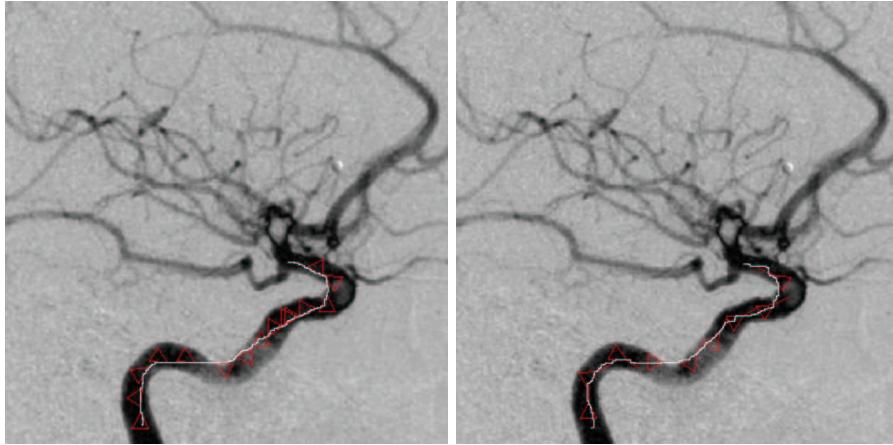


Figure 2.12. Movement of a two different objects in the medium defined by the image on the background: Left image: the object is a triangle; the path represents the trajectory of one of the vertices of the triangle. The constraint on the angle of the object is not sufficient, and the object is doing several U-turns during propagation; right image: the object is now a triangle with a branch connected at one of its vertices; the constraint on the angle of the object is now sufficient, and the object keeps looking in the direction of the trajectory.

2.4.4 Perspectives

Linear objects with self-intersections: in a 2D X-ray image, a linear three dimensional structure projection can self-intersects, like a catheter in a heart image (for example see figure 2.13). The minimal path using only the 2D spatial configuration will not extract the loop which is created. Our method could overcome this drawback.

2.5 Introducing recursivity in the Eikonal equation

The *Fast-Marching* algorithm fails if the penalty is noisy, or if the objects to detect are long thin curves, like the guide-wire shown in figure 2.14. If the offset term w and the penalty \mathcal{P} in *Eikonal equation* are not tuned efficiently, a portion of the shortest path extracted can be a short cut to the starting point, leading to wrong results, like in figure 2.14-left. One way to overcome this drawback is to introduce a recursivity term in the computation in *Eikonal equation*: having $\alpha \in]0; 1]$, we now compute

$$\begin{aligned} & (\max\{u - \alpha \mathcal{U}_{i-1,j,k}, u - \alpha \mathcal{U}_{i+1,j,k}, 0\})^2 + \\ & (\max\{u - \alpha \mathcal{U}_{i,j-1,k}, u - \alpha \mathcal{U}_{i,j+1,k}, 0\})^2 + \\ & (\max\{u - \alpha \mathcal{U}_{i,j,k-1}, u - \alpha \mathcal{U}_{i,j,k+1}, 0\})^2 = \tilde{\mathcal{P}}_{i,j,k}^2 \end{aligned} \quad (2.10)$$

giving the value u for $\mathcal{U}_{i,j,k}$. This recursive term, usually set to .9, reduces the values of the action. This enables the front to propagate further in the direction of the thin



Figure 2.13. Catheter in X-Ray image of the heart: The catheter is the linear structures in dark that self-intersects.

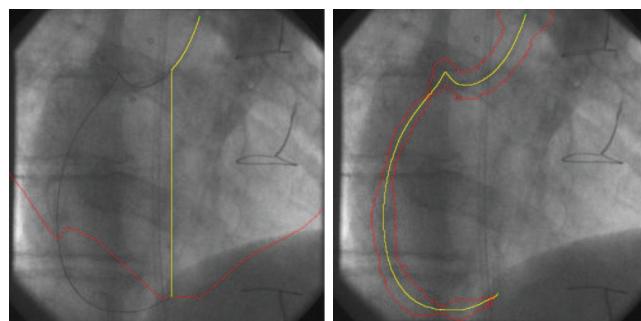


Figure 2.14. Guide-wire extraction with recursive *Fast-Marching* : The left image is the minimal path extraction with classical *Fast-Marching* ; right image is the same result with recursive *Fast-Marching* .

curves, without propagating in all directions. In figure 2.14, the border of the set of visited points is drawn in red: on figure 2.14-left the algorithm has visited more than half the image domain, leading to a wrong path which goes straight down to the end point; on figure 2.14-right, the corresponding domain surrounds the guide-wire, and leads to a path that stay in the vicinity of the object.

Unfortunately, the equation(2.10) of course no more gives the solution to any *Eikonal equation* computed on a penalty domain \mathcal{P} , and the new action map U computed is not convex at all. The minimal path that links the extremities is here defined by a L_1 descent on the map which stores the *Fast-Marching* iterations, and not with the gradient descent on the action map. The minimality principle is lost, whereas this algorithmic trick improves extraction. A patent has been filed on this subject (see [54]).

Chapter 3

Application à l'Endoscopie Virtuelle et à Plusieurs Problèmes en Imagerie Médicale

Abstract — Dans la première partie de ce chapitre on s'intéresse à la construction d'une méthode d'extraction automatique de chemins pour l'Endoscopie Virtuelle. C'est un procédé de navigation dans des images 3D, qui nécessite la définition d'une trajectoire précise pour l'observation en image de synthèse de l'intérieur du corps humain. Nous avons appliqué notre méthode d'extraction de chemins minimaux du chapitre 2 pour construire ces trajectoires rapidement et de manière la plus automatique possible.

La seconde application concerne, au contraire, la construction de chemins de manière interactive, permettant à un utilisateur de dessiner rapidement les contours d'un objet, en ne précisant qu'un point de départ, sur le modèle des techniques appelées *Live-Wire* de *Falcao et Udupa* [49] ainsi que de *Mortensen et Barrett* [127]. Sur la base de notre méthode, nous avons développé un outil similaire, incluant une possibilité d'adapter les paramètres du modèle, au cours de la segmentation, en l'entraînant à reconnaître les contours qui nous intéressent.

Abstract — First section concerns the creation of a fully automatic path tracker for *Virtual Endoscopy*. *Virtual Endoscopy* visualizes the inner surfaces of structures present in volumetric data in 3D images. As navigation through the inner structures quickly becomes a complicated procedure, especially when these structures are strongly curved, often a trajectory through the structure is used. We applied our path extraction technique developed in chapter 2 in order to build an accurate and fast tool to automatically builds those trajectories.

For the second application, we have focused on the need in many applications, as in medical imaging, for interactive segmentation, offering the possibility to a non-expert to draw quickly the boundary of an object, following *Live-Wire* technique of *Falcao and Udupa* [49], and *Mortensen and Barrett* [127]. *Live-Wire* methods restrict the intervention of the user to the setting of a start point in an image. Using our path extraction algorithms, we have developed the same tool, including a training method that adapt the different parameters of the model *On-The-Fly*.

3.1 Virtual Endoscopy

Visualization of volumetric medical image data plays a crucial part for diagnosis and therapy planning. The better the anatomy and the pathology are understood, the more efficiently one can operate with low risk. Various post-processing techniques have been available, to enable the radiologist to recognize a pathological condition, in the shortest amount of time: three 2D orthogonal views (see figure 3.1), maximum intensity projection (MIP, and its variants, see figure 3.2), surface and volume rendering.

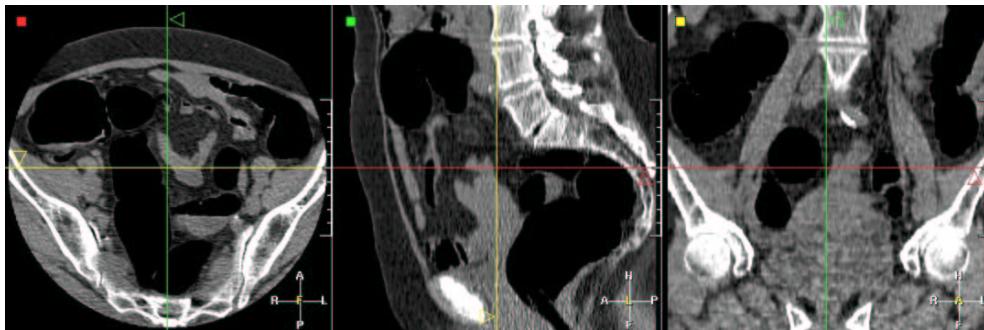


Figure 3.1. 3D CT scanner of the colon: Display of the dataset of $512 \times 512 \times 121$ voxels using three orthogonal views; air was injected in the colon (dark regions of the image) before acquisition in order to inflate the object and increase accuracy of the reconstruction of the colon.

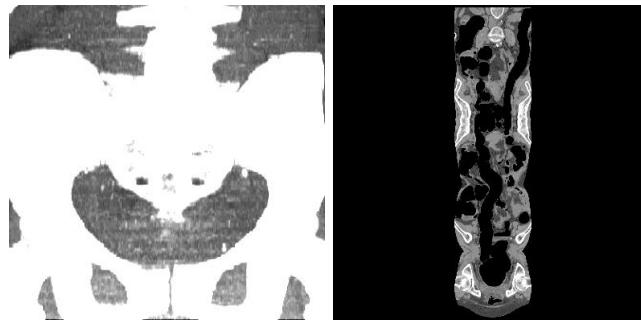


Figure 3.2. Different rendering techniques of a 3D CT scanner of the colon: left image is a Maximum intensity projection MIP and right image is a curved reformat image of the dataset shown in figure 3.1.

The maximum intensity projection requires to chose a direction of projection, therefore the volume is projected on a 2D plane, keeping only the brightest image points (in figure 3.2-left, the brightest intensities are given by the bones).

The curved reformat image in figure 3.2-right is a particular case of a multi-planar reformat image **MPR**: A **MPR** is a cross-sectional image of the 3D volume, along

an arbitrary plane, and a curvilinear reformat is a section along a surface generated by a bi-dimensional curve $f : \mathbb{R} \rightarrow \mathbb{R}, y = f(x), \forall z \in \mathbb{R}$. For a 3D trajectory, there are 3 different ways to represent the **MPR**, by fixing either x , y or z . The path is placed in the target of investigation (the colon in figure 3.1), and the organ appears in the reconstruction in its full length. Advantage is to display the whole complex shape of the colon on a single 2D image, but the anatomical objects are distorted, and dimensions are not reliable on the final image.

Virtual Endoscopy allows by means of surface/volume rendering techniques to visually inspect regions of the body that are dangerous and/or impossible to reach physically with a camera (e.g behind an airway stenosis or obstruction, or too small). An extensive definition of *Virtual Endoscopy* can be found in [80, 146]. In chapter 7, we detail visualization techniques based on volume and surface renderings.

Virtual Endoscopy techniques can be divided into two groups of methods that can collaborate:

- techniques which deal with simulation of a real endoscope motion; In this case, *Virtual Endoscopy* is very interactive, simulating the motion of a camera inside the body, based on an extracted anatomical object that is modeled using rigid body dynamics; good examples of this simulation are presented in [77, 131].
- techniques which focus on the observation of the interior of anatomical objects by extracting trajectories inside them, see Yeorong *et al.* [192] for an example.

We have focused on the second kind of techniques. However, the minimal path techniques can also be useful for the first kind of methods: Kimmel and Sethian have applied the Fast-Marching algorithm for a robotic application in [163], for the motion of an object with a certain shape and orientation in an image with obstacles. We have also investigate this field in section 2.4, modeling the movement of an object, discretizing *Eikonal equation* in a space that describes the object position and orientation. But adding a dimension to the problem could lead to huge computing costs for an interactive 3D application.

Figure 3.3 is the usual framework that builds a *Virtual Endoscopy* facility, and is usually composed of two parts:

- A Path construction part, which provides the successive locations of the fly-through in the tubular structure of interest (see figure 3.3-left);
- Three dimensional interior viewing along the endoscopic path. Those views are adjoined creating an animation which simulates a virtual fly-through through them (see figure 3.3-right).

3.1.1 Medical relevance

In recent years, computerized post-processing techniques of image data from cross-sectional imaging modalities has received increasing recognition in the field of medicine. Technical developments of acquisition systems such as CT and MRI have improved along with continuously increasing spatial resolution. But if each axial image were

44.3 Application to Virtual Endoscopy and to Several Problems in Medical Imaging

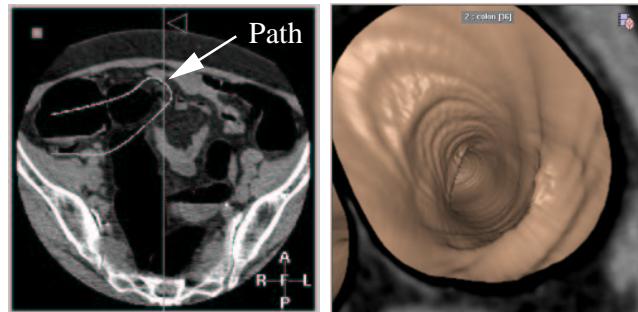


Figure 3.3. Virtual Endoscopy framework: Left image shows an example of trajectory manually drawn using the orthoviewer facility; right image is a volume rendering of the dataset, using a ramp function manually parameterized, at a position along this path.

to be viewed before one could proceed to the next image, the viewer would require an enormous amount of time to shift through all slices. At the same time, growing computer performance has created the opportunity to display anatomical structures in a comprehensible manner. *Virtual Endoscopy* is one of the most recent innovations in the spectrum of post-processing techniques. The predominant motive is to simulate conventional endoscopy, as in figure 3.4-left, by means of a non-invasive and safe technique, presenting the image data included in the original slices, in a movie-mode, in such a fashion that the radiologist is able to differentiate between that which is healthy and that which is pathological. Figure 3.4 shows how a clinician is able to detect pathologies using the volume rendering tool associated to the centered path extraction. But the true diagnostic performance of *Virtual Endoscopy* and the ability

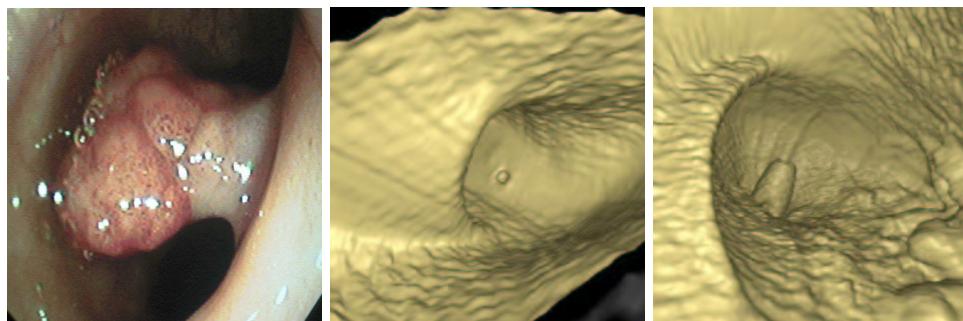


Figure 3.4. Pathologies detected with the volume rendering: Left image is a conventional endoscopy image of the colon; the middle image displays a small cavity which is a diverticulum in the colon surface; on the lower-right part of the right image, the bumps are polyps.

to reproduce acceptable results outside of research protocol has not been established yet. Clinical validation is still an on-going process, and researchers focus on the ap-

plication and validation of this technique for particular organs and pathologies. For colorectal cancer (CRC) (see figure 3.4-right), studies (see [146]) show that net effect of the use of *Virtual Endoscopy* could reduce cancer-related deaths in the future.

3.1.2 Problem with the path construction

A major drawback in general remains when the path construction is left to the user who manually has to “guide” the virtual endoscope/camera. The required interactivity can be very tedious for complex structures such as the colon for example. Actually, on most clinical platforms the user must define all path points manually, using for example three 2D orthogonal views, as shown in figure 3.1, leading to problems as the following:

- Since the anatomical objects have often complex shapes, they tend to pass in and out of the three orthogonal planes. Consequently the right location is accomplished by successively entering the projection of the desired point in each of the three planes;
- The path is approximated between the user defined points by lines or Bezier splines. If the number of points is not sufficient, it can easily cross an anatomical wall.

Path construction in 3D images is thus a very critical task and precise anatomical knowledge of the structure is needed to set a suitable trajectory, with the minimum required interactivity.

Numerous techniques try to automate this path construction process. Most of them use a skeletonization technique, like in [192], in order to extract a centerline in the dataset. But extracting the skeletons of an anatomical shape requires first to segment it. And the skeleton often consists in lots of discontinuous trajectories, and post-processing, as done by Tek and Kimia [173] is necessary to isolate and smooth the final path. The front propagation techniques studied in this application in contrast to other methods does not require any pre- or post-processing as explained in section 2.3.

It is sometimes necessary to smooth the path extracted by the front propagation. The point of view in the volume rendering of the tubular structure is very important, because it constrains the result of the examination. Thus, during the virtual fly through, the point of view of the camera must change smoothly. Traditionally, the position of the virtual camera frame at a particular path point is orthogonal to the path. If the path is not smooth, the point of view of the virtual camera will change in an abrupt manner. There are two ways to achieve this regularization:

- by modifying the view angle of the virtual camera, being no more orthogonal to the path, but looking in the direction of a path point which is located far from its current position, or using a running average of the local direction of the camera;
- by increasing the weight w in equation (1.3) since it has a smoothing effect on the minimal path (see appendix for details). We preferred to use this technique in the following examples, since it is efficient and very simple to add.

3.1.3 Proposed solution for colonoscopy

The method detailed here is illustrated by results performed on the volumetric CT scan shown in figure 3.1.

Building a potential for virtual colonoscopy

The target is to build a potential P with the 3D data set allowing paths to stay inside the anatomical shapes where end points are located. We thus define the potential by a general model $\tilde{P}(x) = |I(x) - I_{mean}|^\alpha + w$.

First, the potential must be lower inside the colon in order to propagate the front faster, and to avoid problems with crossing the edges of the anatomical object. In a colon CT scan, an average position I_{mean} of the colon grey level in the histogram can be defined (see figure 3.5) as a peak in the histogram where $I_{mean} = 200$. Secondly,

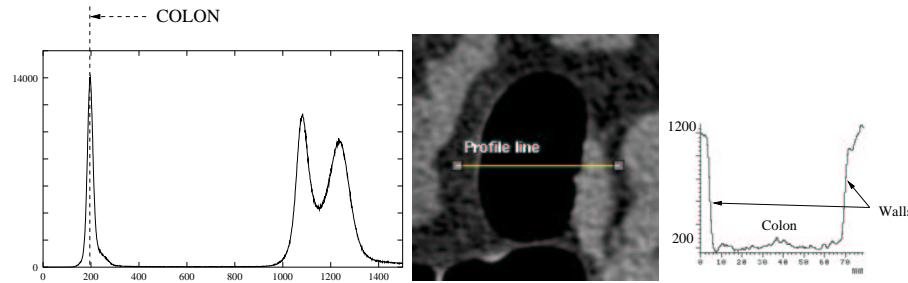


Figure 3.5. Global and Local study of the image intensities: Left image is the histogram of the whole 3D dataset where the colon can be clearly distinguished; middle image is a slice of the colon where a profile line was drawn in order to observe the variation of intensities that are displayed in right image.

if the path to be extracted is very long, the situation can lead to pathological cases, and the front can go through potential walls. This is frequent for large objects that have complex shapes and very thin edges, as colon. Then, edges should be enhanced to enable long trajectories, with a non-linear function. We thus take $\alpha = 2$ in order to enhance the dynamic of the image with a quadratic function.

However, this potential does not produce paths relevant for *Virtual Endoscopy*. Indeed, paths should remain not only in the anatomical object of interest but as far as possible from its edges. In order to achieve this target, we use the centering potential method as detailed in section 2.3. We first need to obtain a shape information. In fact, a CT scan of the colon contains already a shape information sufficient to constrain a front propagation. In figure 3.5-middle is shown a slice of a colon volumetric data set, and figure 3.5-right shows the corresponding grey level profile. Air fills the colon and is represented in our CT image by a grey level around 200 (see figure 3.5-right), while edges are defined by a grey intensity around 1200. Then, using the potential $\tilde{P}(x) = |I(x) - I_{mean}|^\alpha + w$, the front obtained through *Fast Marching* is stopped by the anatomical shapes. Figure 3.6 shows the propagation of the wave equation, according to the penalty defined previously. It also illustrates the fact that the *Fast-Marching* can act also as a segmentation tool.

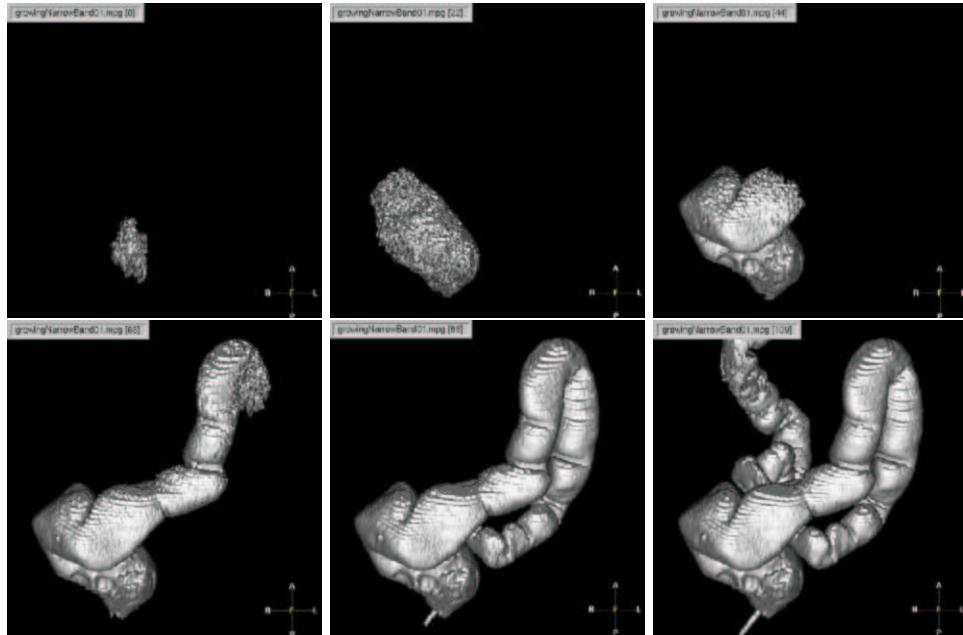


Figure 3.6. Wave propagation inside the colon dataset: these are volume renderings of the *Alive* points at different consecutive iterations during propagation.

Path centering technique

The edges are obtained via a first propagation: in figure 3.6 we can see the evolution of the *narrow band* during propagation. It gives a rough segmentation of the colon and provides a good information and a fast re-initialization technique to compute the distance to the edges. Using this distance map as a potential (from equation (2.7)) that indicates the distance to the walls, we can correct the initial path as shown in figure 3.7-left: the new path remains more in the middle of the colon. And the value of the parameter d can be derived from anatomical characteristics. If we know approximately the section of the colon along the path we can easily choose a value to stay in the center of the tubular structure.

The two different figures 3.7-middle and 3.7-right display the view of the interior of the colon from both paths shown in figure 3.7-left. With the initial potential, the path is near the wall, and we see the u-turn, whereas with the new path, the view is centered into the colon, giving a more correct view of the inside of the colon. The new centered path is smooth because this final propagation is done on a synthetic potential (the distance to the walls) where noise has been removed.

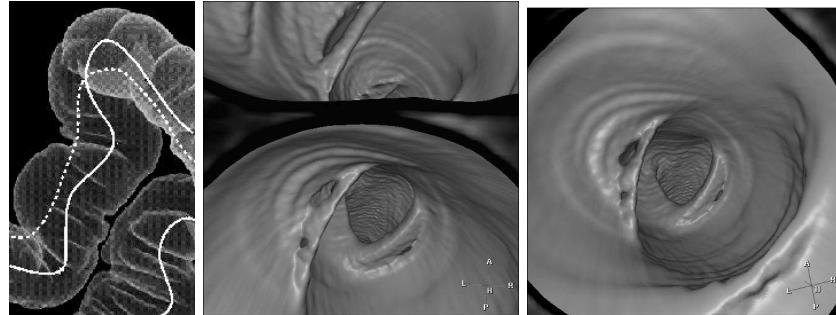


Figure 3.7. Comparing normal and centered paths: The left image is a Tissue Transition Projection (TTP) of the colon surface; middle image is the endoscopic view obtained with the classical trajectory (represented with a dotted curve) in the U-turn shown in the left image; right image shows the endoscopic view resulting from the centered path (represented with a plain curve on left image) at the same position in the colon.

3.1.4 Results on objects filled with air

Results on Colonoscopy

Virtual colonoscopy is one of the most exciting developments in gastrointestinal radiology. It is a non-invasive, well-tolerated, and safe technique for evaluating colorectal cancer (CRC). CRC represents the third most frequently diagnosed cancer worldwide. For the United States, it is estimated that 129,000 new cases were diagnosed in 1999 [168]. Preliminary results indicate that the accuracy of virtual colonoscopy exceeds that of conventional endoscopy. Our automatic path extraction provides in a very fast process a path that remain inside the structure and avoid collisions with the wall, following in a smooth manner the centerline of the colon. Once this path is created, navigating is simply a matter of positioning the view along the path. Figure 3.8 shows the difference between the paths extracted with the classical method, and the centering method.

Notice that in figure 3.8-middle, the path has been smoothed by the centering method. This is mainly due to the fact that the noise inside the colon has been eaten by the first propagation. The small variations in intensity allows the front to propagate in all the colon, and considering the distance to the borders of the object *cleans* the anatomical object, by considering an artificial penalty $P(x, y, z) = 1 \forall (x, y, z) \in \mathbb{N}^3 \cap \text{colon}$. Figure 3.8-right illustrates the complexity of the path extracted: intersecting the same slice several time, the user who wants to extract it manually must have strong skills in anatomy, and time. Figure 3.9 shows the corresponding endoscopic viewings generated with those two paths.

Results on the Trachea

The virtual inspection of the trachea is the same problem as in the colon. It is even simpler, due to the topology of the organ observed. Figure 3.10-left displays a 3D

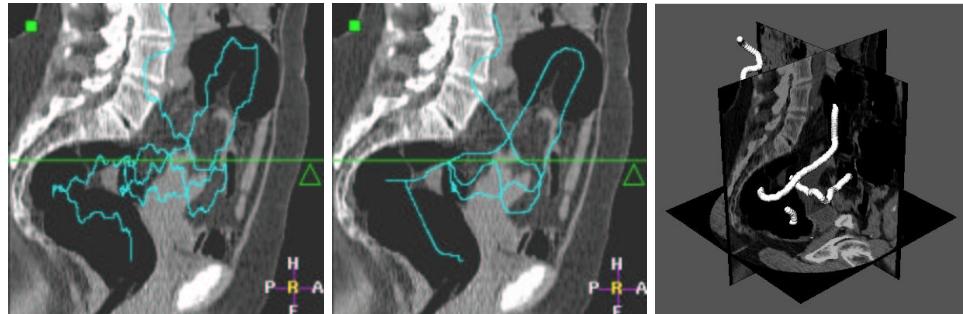


Figure 3.8. Path extraction in the 3D CT scanner of the colon: Left image is the superposition of a path extracted from only one point manually located inside the colon over a slice of the dataset; middle image displays the centered path in the same configuration; right image is the representation in 3D of the trajectory displayed in middle image intersecting three planes, where the dataset has been mapped.

CT dataset of a trachea with three orthogonal views. Air fills the object and gives a shape information all along from throat to lungs. Therefore, the anatomical object having a very simple shape, the path construction with one or two fixed points is easier than in the colon case. One example path tracks the trachea, using a nonlinear function of the image grey levels ($\tilde{P}(x) = |I(x) - 200|^2 + 1$). This path has been used to display the **MPR** view of figure 3.10-right. An endoscopic view along the same path is displayed in figure 3.11. The inspection of the trachea is often part of the inspection of the whole tracheobronchial tree. In part III, we will study more precisely the tracheobronchial tree. The examination of the complete tree needs new algorithms to extract the complete tree structures, and to segment the borders of the bronchi. This implies use of more complicated algorithms than *Fast-Marching* and will be further developed.

3.1.5 Results on Arteries and vessels

Is endoscopy a useful tool for artery and vessel examination?

The volume-based rendering tool use an opacity threshold which can create severe artifacts on the endoscopic viewings. And the quality of the images is a direct result of the homogeneous opacification of the blood. The blood is made visible using contrast products, and this image information depends now on the section of the object. It can be also severely impaired in areas of turbulent flows.

Review of the axial slices alone is somehow sufficient for the diagnosis assessment of several pathologies. But, still automatic path extraction provides important visualization assessments on the datasets. It can be used with multi planar reformatting **MPR** images, which enable to see the tubular objects with optimal slice orientation, depicting spatial relationships between the object and its pathologies. But *Virtual Endoscopy* offers an intraluminal view of the arteries and veins and allows inspection of pathologies, like stenosis. It enhances visualization, by clearly showing the spatial

50 3 Application to Virtual Endoscopy and to Several Problems in Medical Imaging

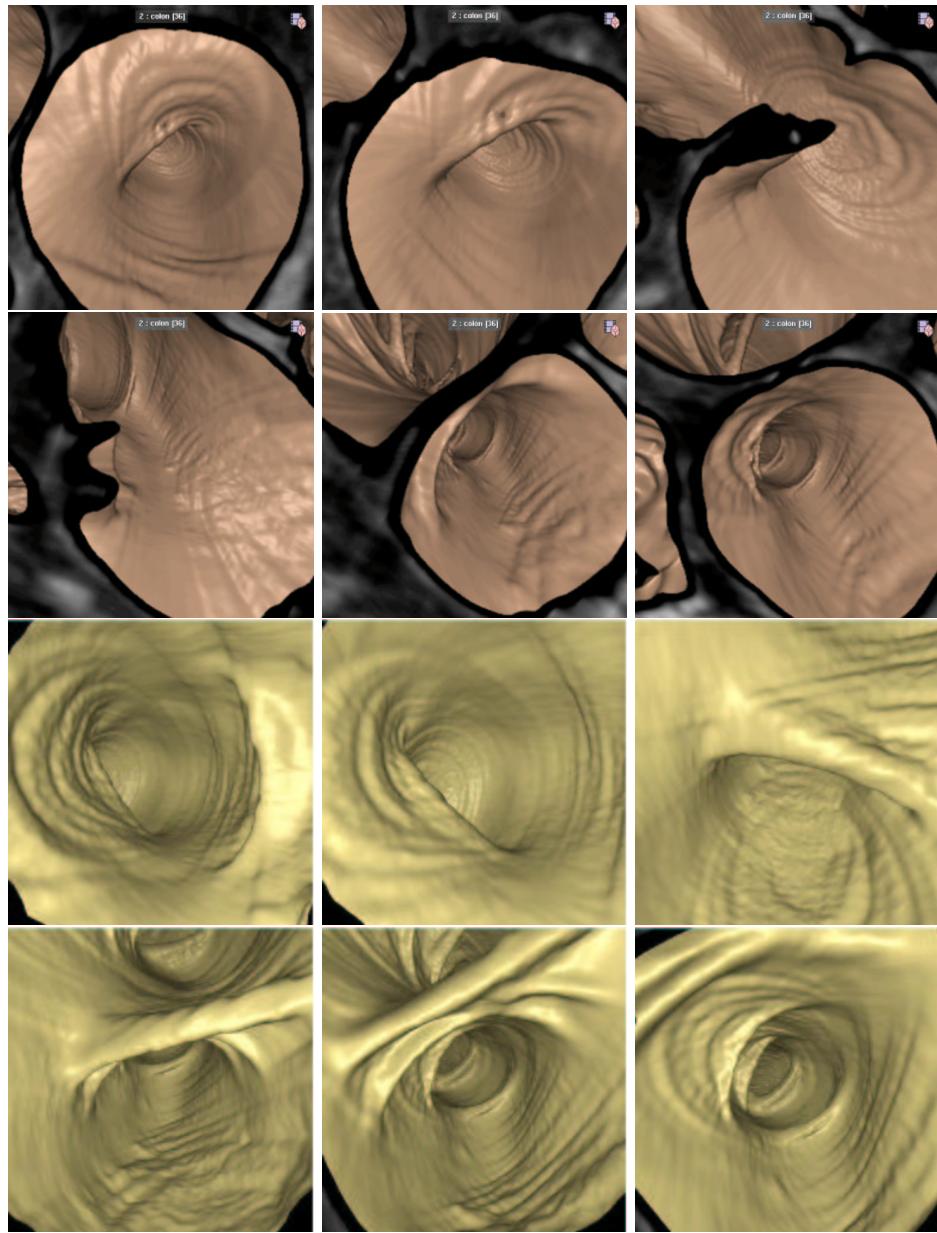


Figure 3.9. Virtual Endoscopy in the colon, minimal and centered paths:
On the first two rows the trajectory being the shortest, the intersection plan of the camera shows adjacent structures and shrinks the validity of the examination; on the two last rows, the trajectory being centered, the point of view of the virtual camera shows the entire tube, and the validity of the examination is increased.

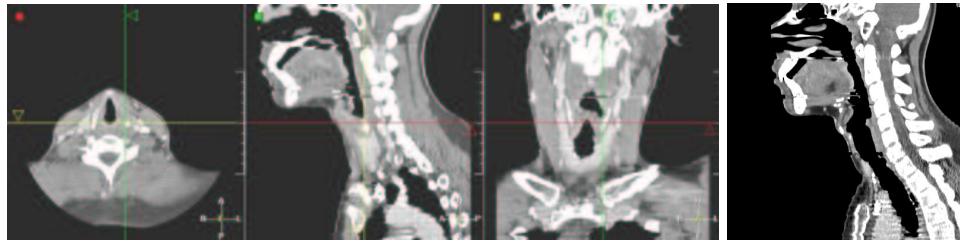


Figure 3.10. 3D CT scanner of the trachea: Left image is a dataset of $320 \times 320 \times 234$ voxels; the interior of the trachea is very easy to characterize from the whole image since it is always filled with air; right image is a **MPR** image along the trajectory extracted.

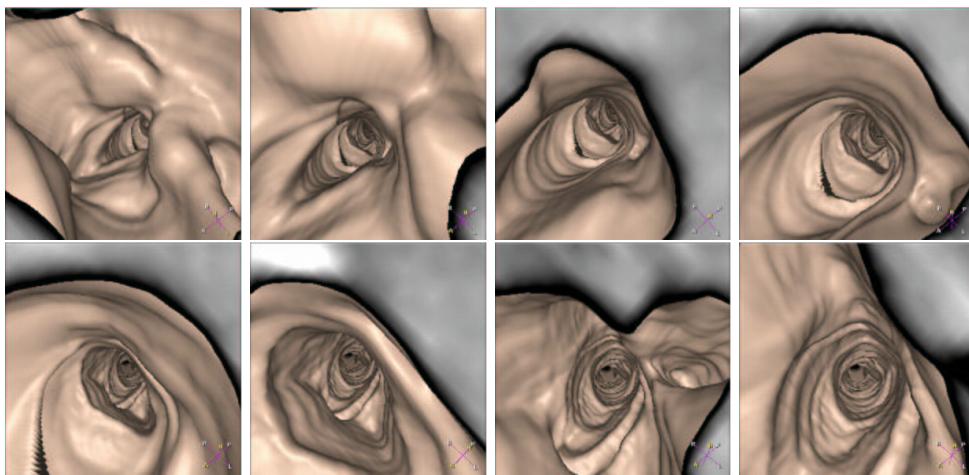


Figure 3.11. Virtual Endoscopy in the Trachea: starting from the mouth, the virtual camera goes straight to the bifurcations between the lungs, following the path extracted (see **MPR** view in figure 3.10-right).

relationship between the pathology and the object, comparing those pathologies before and after treatment, as done for stent placement in abdominal aortic aneurysm **AAA**. It can be also input in systems for the virtual planning of endoluminal aortic stents (see [186]).

This gives justification for the following study on the automatic path extraction in veins and arteries. But the grey-level information is more complex in the case the signal is obtained through injection of a dye product in the object of interest. The boundaries of the object are more difficult to extract, therefore the path centering technique developed in section 2.3 and previously applied to the colon case, is no longer valid for the following. However, automatic path extraction has remained valid, despite the variability of the new penalty information. Centering techniques for those kind of objects involve the use of more complex algorithms, like achieved segmentation algorithms (see *McInerney and Terzopoulos* [115] for a review of med-

ical image segmentation with deformable models), and have been later developed *a posteriori* in part III of the thesis.

3D MR image of the Aorta

A test was made on an aorta MR dataset. Figure 3.12-left displays this 3D MR dataset of the abdominal aorta using three orthogonal views. Contrast product was injected

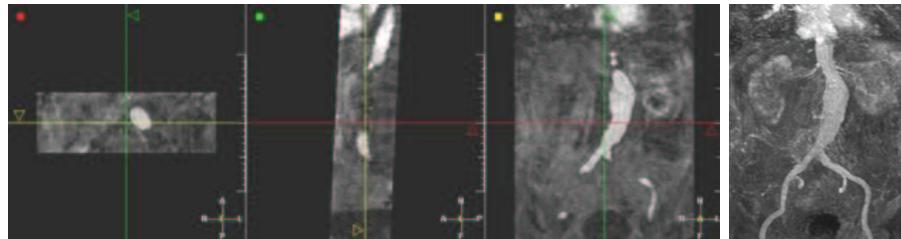


Figure 3.12. 3D MR image of the aorta: Dataset of $256 \times 256 \times 60$ voxels; a dye product has been injected before acquisition to highlight the abdominal aorta; the right image is a threshold based volume rendering of the aorta itself, which highlights that the anatomical object has a visible pathology: an Abdominal Aortic Aneurysm (AAA).

before acquisition. On the **MIP** view, in figure 3.12-right, the important variation of the section of the object, upon the bifurcation of the iliac arteries, clearly indicates an an Abdominal Aortic Aneurysm **AAA**. The dye fills the aorta, and makes it visible, among other soft tissues, while bones are not rendered. The propagation measure is based on a nonlinear function of the intensity of the contrast solution that fills the aorta. This data set is difficult since the intensity of the contrast product will vary along the aorta (the contrast bolus dilutes during the acquisition time). Due to this non-uniformity, paths can cross other anatomical structures with similar intensities if the mean value inside the aorta is not set correctly by the user. Our example path tracks one iliac artery (see figure 3.13), using the potential $\tilde{P}(x) = |I(x) - 1000|^2 + 10$ in the MR scan. The dataset contains noise, and we must use an important weight to smooth the extracted paths. We have displayed a sample of the endoscopic views of the aorta along the path in figure 3.14. During the virtual fly through, the observer clearly notice the important variation of the object cross-section. The movie in figure 3.14 depicts this pathology, even if the threshold-based volume rendering produces artifacts, when using the image grey level information obtained from a contrast enhanced MR image.

3D CT Scanner of the Aorta

Figure 3.15 displays a 3D CT dataset of the abdominal aorta using three orthogonal views. The contrast product injected before acquisition fills the aorta, and make it visible among other objects. The problem is similar to path extraction in a MR image, as done previously. Main difference here lies in the high resolution of the CT scanner

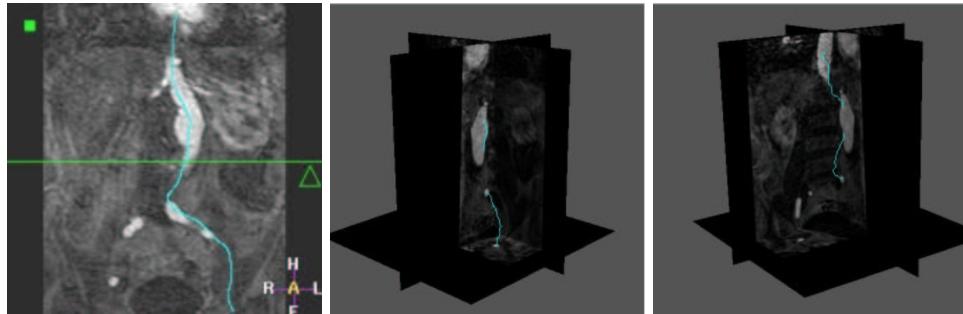


Figure 3.13. Path extraction in the 3D MR image of the aorta: Left image is the superposition of one path extracted between two points manually located inside the aorta over a slice of the dataset; middle image is a curved reformat sagittal image along one of the paths which tracks the right iliac artery; right image is a volume rendering view based on an opacity function parameterized manually along the same trajectory.

dataset of figure 3.15, towards the MRA dataset of figure 3.12. This high resolution increases significantly the computing time of the method, but even if the dynamic of the images is very different, it does not lead to major differences in the parameterization of the path extraction process. Figure 3.16-left displays several paths extracted with the same seed point. Figure 3.16-middle is a curved reformat view along one of the trajectories extracted. This kind of view enables to validate trajectories by verifying that the section drawn always intersect the structure of interest.

3D MR image of the brain vessels

Tests were performed on brain vessels in a MRA scan. Three orthogonal slices of this dataset are shown in figure 3.17 together with a path extracted. The problem is different, because there is only signal from the dye in the cerebral blood vessels. All other structures have been removed. The main difficulty here lies in the variations of the dye intensity. The example path tracks the superior sagittal venous canal (the vein on the top of the head, as shown in figure 3.17-right), using a nonlinear function of the image grey levels ($\tilde{P}(x) = |I(x) - 100|^2 + 1$). Two views of the extracted path in 3D are displayed in figure 3.18 together with 3 orthogonal slices of the dataset. A sample of the virtual fly-through along the brain vessel is displayed in figure 3.19.

3.1.6 Clinical study

Goal

A multi-user clinical study was performed using a prototype based on EasyVision (Philips Medical Systems). The purpose was to measure the speed and user-dependence of the automatic path tracker. To this aim, the path tracking tool has been evaluated by 5 different operators :

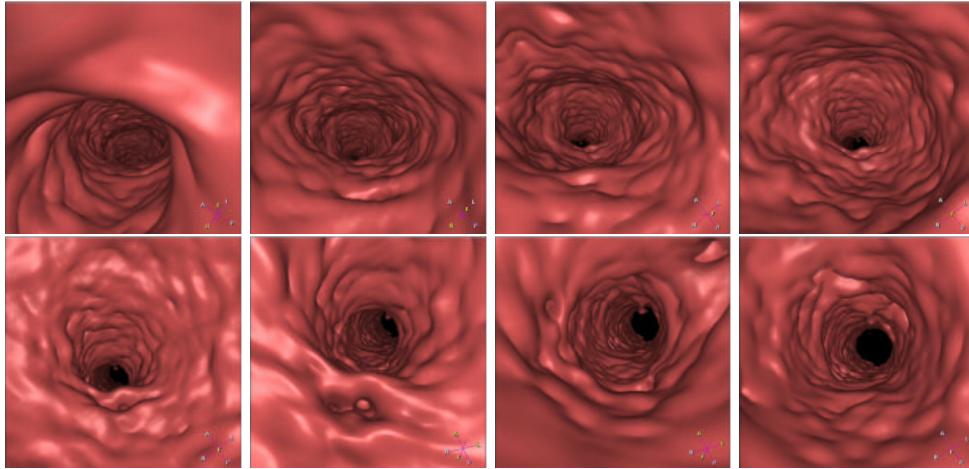


Figure 3.14. Virtual Endoscopy in the Aorta: starting from one iliac artery, the virtual camera goes to the top of the aorta, and the section of the object becomes larger when it is located inside the aneurysm.

- two physicians with manual path tracking experience (P_1, P_2);
- two operators with abdominal anatomy knowledge but no virtual colonoscopy practice (M_1, M_2);
- one reference operator (R) familiar with the automatic path tracking tool.

As a comparison the user R also manually defined a path twice on the same dataset.

Data

Spiral CT data (5 mm slice thickness, 3 mm reconstruction interval) from 15 patients were used, corresponding to a total of 29 scans. During the patient preparation phase the colon was emptied as much as possible, and distended by inflating room air. In most cases, both prone and supine scanning was done.

Measurements

Time For each user, the wall clock time was measured using an automatic logging mechanism. In general, path construction consisted of following steps :

1. load and inspect data
2. place starting point in the cecum
3. track + center path
4. check result and modify/continue tracking if necessary

The time necessary to perform steps 3 and 4 were measured and both user interaction time and calculation time were taken into account.

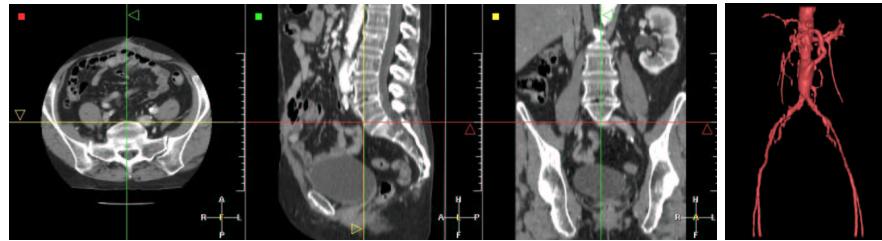


Figure 3.15. 3D CT Scanner of the aorta: Dataset of $512 \times 512 \times 173$ voxels; a dye product has been injected before acquisition to highlight the abdominal aorta; the anatomical object does not have a visible pathology; the right image is a threshold based volume rendering of the aorta itself (the MIP view is disturbed by the intensities of the bones).



Figure 3.16. Path extraction in the 3D CT scanner of the aorta: Left image is the superposition of several paths extracted from the same seed point at the top of the aorta, over one slice of the dataset; middle image is a curved reformat view along one of the trajectories extracted; right image is an endoscopic view along this path.

User-dependence To measure the user-dependence of the path tracker, resulting paths P from different users were compared to the corresponding path R obtained by reference user in the following way :

1. **Warp path $P(i)$ and $R(j)$ to a common length parameter k**
We want to locally stretch and compress paths $P(i)$ and $R(j)$ using warping functions $w_P(k)$ and $w_R(k)$. The goal is to obtain the optimal warping satisfying

$$(w_P, w_R) = \arg \min_{w_1, w_2} \left(\sum_k \|P(w_1(k)) - R(w_2(k))\| \right) \quad (3.1)$$

where all points of P and R are addressed by the warping. This mapping is calculated using the Dynamic Time Warp algorithm [145].

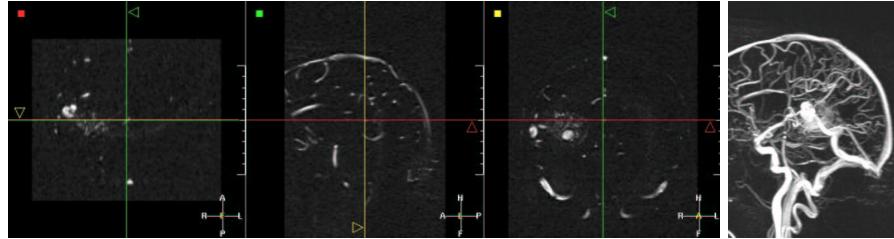


Figure 3.17. 3D MR Angiography image of the brain vessels: Dataset of $256 \times 256 \times 150$ voxels; It is obtained by subtracting two acquisition: one before, and one after injection of a dye product; thus there is only signal coming from moving objects; right image is a MIP view of this dataset.

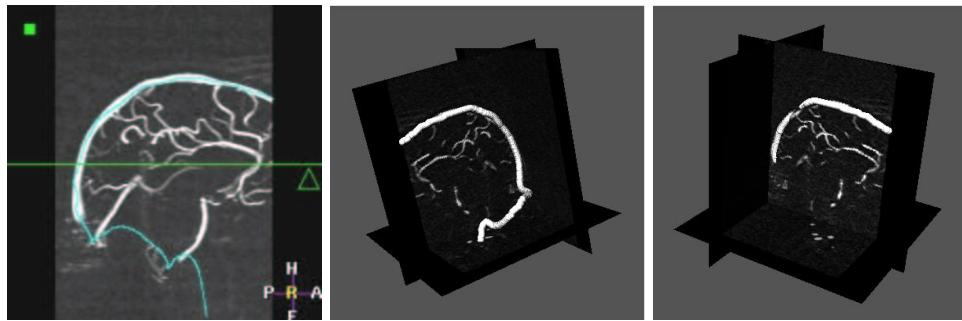


Figure 3.18. Path extraction in the 3D MR Angiography image of the brain vessels: Left image is the superposition of one path extracted between two points manually located inside the superior sagittal sinus over a slice of the dataset; middle and right images are the representation in 3D of this trajectory intersecting three planes, where the dataset has been mapped.

2. Calculate Euclidean distances d between corresponding points

After warping, the path distances d can be calculated as

$$d_{P,R}(k) = \|P(w_P(k)) - R(w_R(k))\| \quad (3.2)$$

and are shown in Fig.3.20 as a function of the common path length k .

3. Extract the common part

To exclude the effect of the exact start and end point, we only consider the common part of paths P and R , which is defined in Fig.3.20.

4. Measure the similarity S between the paths

We propose to measure the similarity by using the percentage of the common path length k where the distance d is below a threshold t , written as $S_{(P,R)}^t$.

The chosen measure for path similarity $S_{(P,R)}^t$ is symmetrical, it is robust to a complete different choice in starting position, and it gives a more reliable result of

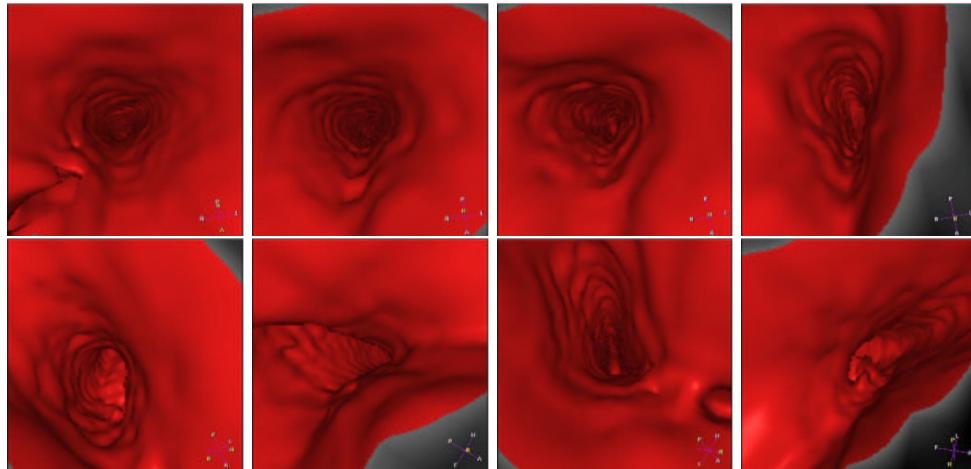


Figure 3.19. Virtual Endoscopy in a brain vessel: the virtual camera goes into the superior sagittal sinus venous canal.

similarity between paths than e.g. the maximum distance.

Results

Time The results of the time measurements are shown in Table 3.1. The average time needed for path tracking is 4.8 minutes per scan, measuring both user interaction time and calculation time.

time	P1	P2	M1	M2	R	average	manual
user time	2.8	4.6	5.3	4.5	2.0	3.6	30.0
calculation time	1.0	1.0	1.2	1.7	1.3	1.2	
total time	3.8	5.6	6.5	6.2	3.2	4.8	30.0

Table 3.1. Timing results of the automatic path tracker: Time expressed in minutes per scan.

No significant differences were found between the experienced physicians (P_1 and P_2) and the other users (M_1 , M_2), excluding the reference user R . These results are in agreement with a previously published study [147], where an average of 4.5 minutes was measured on 27 cases. The timing for the manual case has no statistical meaning, but is merely given for comparison.

User-dependence Table 3.2 summarizes the measurements of the path correspondence using the similarity measures S^2 (2 mm threshold) and S^5 (5 mm threshold) as defined in section 3.1.6.

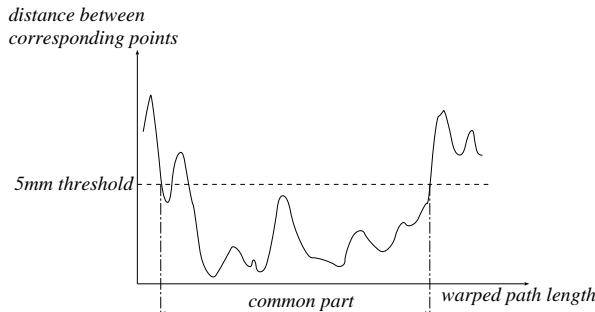


Figure 3.20. The Euclidean difference between corresponding points plotted versus the warped path length k . The common part is defined as the longest possible stretch between two positions where the distance falls below the threshold of 5 mm.

correspondence	P1	P2	M1	M2	average	manual
S^2	79%	89%	81%	92%	85%	20%
S^5	89%	97%	93%	95%	94%	68%

Table 3.2. Correspondence with reference path: Correspondence expressed in percentage of the path length where the reference path is closer than 2 mm (first row) or 5 mm (second row).

On the average, an automatically defined path differs less than 5 mm from the reference over 94% of its length. Again, the results of the manual path tracking is given for comparison. Both manually tracked paths differed less than 5 mm over only 68% of their lengths.

Automatic path tracking provides a fast and easy way to determine the colon centerline. The resulting path lies completely inside the colon, is as much centered as possible and is smooth.

The path tracker can be used by less experienced operators without significant differences in time or resulting centerline. This is an important result, since it allows to separate the path tracking task from the actual inspection task. The former task can be done as a preprocessing step by a different person since the results of the path tracker are largely operator-independent. Separating path tracking and inspection will increase the physician's efficacy, reduce the cost and allow a more widespread application of path-based navigation and visualization.

3.1.7 Last developments and perspectives

Figure 3.21 shows how the colon surface is unfolded using the point of view of the virtual camera which is given by the centered path extracted. This unfolding¹ enables to clearly see at each path position desired the rendering on each side of the camera

¹This image was provided by Roel Truyen, from MIMIT Advanced Development, Philips Medical Systems, Best, Netherlands.

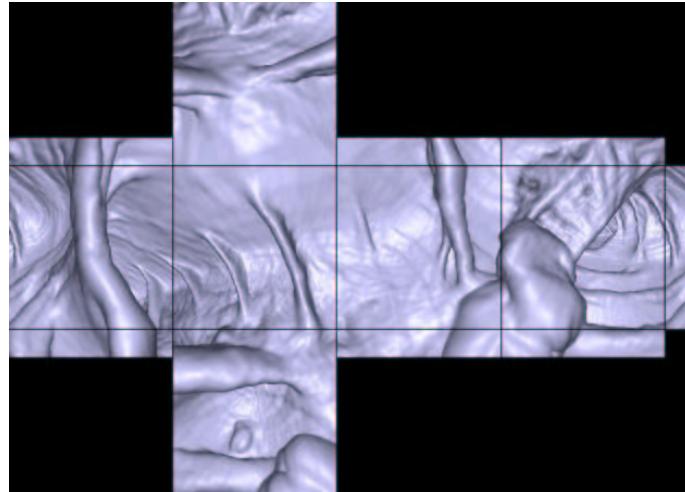


Figure 3.21. Unfolding the colon surface:

and behind, thus increasing the accuracy of the diagnosis. For example, in the lower part of figure 3.21, the bump on the colon surface is a polyp that would not be detected if the point of view given follows the trajectory direction. Therefore, the only requirement for an accurate fly-through is to provide a trajectory that follow the centerline of the structure, as closely as possible, as this provides the best visualization of the surrounding walls. The path should also be smooth without unnecessary kinks.

Further work on the subject of endoscopy could be to simulate directly the trajectory of an endoscope inside the anatomical objects. The introduction of the angle as a dimension in *Eikonal equation* in section 2.4, and the algorithmic tricks developed to compute a minimal action for a moving object in the same section, could lead to the extraction of minimal paths in the human body for moving objects with a given shape. A straightforward application of this angular propagation for an object trajectory is to guide objects in a virtual endoscopic process. If the object is not a point, the modification of its orientation will now represent a cost to minimize. If this cost is also related to the refraction indices of the medium, the constraint will regularize not the trajectory of the object, but its orientation. For a virtual endoscopic camera, regularizing the point of view will lead to a better understanding of the scene and of the pathologies. Of course, the computing time is multiplied by the number of discretized angle in $[0; 2\pi]$, thus the problem is now four dimensional, and the path extraction is really time consuming. But we can forecast that the growing computer performance in the next years will make this improvement feasible.

3.2 Live-Wire

Approaches in image segmentation are numerous, ranging from fully automatic methods to fully manual methods. The first ones totally avoid user's interaction but still are

60 3 Application to Virtual Endoscopy and to Several Problems in Medical Imaging

an unsolved problem: even if they are well adapted to specific cases their success can not be guaranteed in more general cases. The second ones are time-consuming, unrepeatable and inaccurate. To overcome these problems, interactive (or semi-automatic) methods are used. They combine knowledge of the user and computer capabilities to provide supervised segmentation, ranging from manual painting to minimal user intervention.

The target of this application was interactive and real-time extraction of features in medical images, independently from any acquisition modality. The aim was to develop a method to offer the possibility to a non-expert to draw quickly the boundary of an anatomical object. He could for example restrict its intervention to the deposition of a start point in an image. Then a contour had to be automatically found and drawn in real-time between this start point and the current cursor position. This contour should respond to a certain amount of constraints, such as internal and external forces and action of the user. The developed method should let the user validate or not the result. Taking this validation into account, the tool should be able to generate a kind of learning to better estimate the different parameters of the model.

In contour oriented segmentation, one approach is to define a boundary as the minimum of an energy function that comprises many components such as internal and external forces. In the literature, there exist many techniques to perform this minimization. First, classical active contours (also called Snakes or Deformable Boundaries), introduced by *Kass, Witkin and Terzopoulos* [82], have received a lot of attention during the past decade. But this technique presents three main problems. First, variational methods are very sensitive to the initialization step and often get trapped in a local minimum. Second, user's control cannot be applied during the extraction but only during the initialization (where the user has a whole contour to draw) and the validation stages of the segmentation. Third, the different parameters of the model are not meaningful enough in a user's viewpoint, especially for clinicians. The application of the minimal path theory to image segmentation is a more recent technique. With this approach the image is defined as an oriented graph characterized by its cost function. The boundary segmentation problem becomes an optimal path search problem between two nodes in the graph. This approach overcomes the problem of local minima by using either dynamic programming (*Dijkstra* [43]), or a front propagation equation (*Cohen and Kimmel* [34]), mapping the non-convex cost function into a convex function. Dynamic programming has also been used for classical snakes [4], but the proposed method is applied there to find the local deformation from an initial curve that gives the best energy descent. *Falcao and Udupa* with their *Live-Wire* [48, 49] and *Mortensen and Barrett* with their *Intelligent Scissors* [129, 126, 127, 128] introduce interactivity into the optimal path approach. Their method is based on Dijkstra's graph search algorithm and gives to the user a large control over the segmentation process. The idea is the following: a start point is selected by the user on the boundary to be extracted, and an optimal path is computed and drawn in real time between this start point and the current cursor position (see figure 3.22). Thus, user's control is applied also during the extraction. *Mortensen and Barrett* [10, 127] also introduce facilities called *Path-Cooling* and *On-The-Fly* training, which respectively lead to the possibility of drawing a closed contour, and to partial adaptation of the graph cost function. This technique seems to be the most

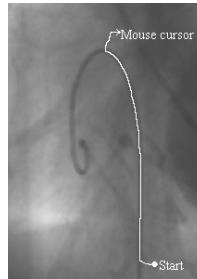


Figure 3.22. Principle of interactive contour extraction with optimal path method: the optimal trajectory is extracted in real time between a user defined starting point and the mouse cursor.

adequate according to our target. The application consisted therefore, first in the implementation of a method based on minimal path search inspired from *Live-Wire* and *Intelligent Scissors* tools, and second in using this implementation to go further with the idea of interaction and learning. The optimal path approach is developed in the first part of this section through the description of *Live-Wire* [49] and *Intelligent Scissors* [127]. The second part explains the adaptation we made of these techniques, including adjustments and improvements.

3.2.1 Existing methods

The aim of this work is interactive segmentation of contours in images. In contour oriented methods, one approach is to define the boundary as the minimum of an energy function, also called cost function. This cost function includes external energy terms describing the salient features that can be extracted from the image and internal energy terms ensuring the regularization of the segmented curve. With the minimal path approach the minimization is not global but local: the aim is to find the optimal boundary segment between two points, that is to say the contour segment where the energy is minimal. This can be achieved using the graph search theory, as detailed in section 1.3.1 .

Two different version of the minimal path extraction were used in the following:

1. first one is the *Dijkstra* algorithm [43], which is detailed in section 1.3.1.
2. second one is the *Fast-Marching* implementation presented in section 1.3.2.

Once the method is chosen, the result mainly depends on the choice for an acceptable cost function.

Cost function

The optimal-path search is guaranteed to find the solution of the minimization of the energy function between two points. But if this function does not pertain enough to the object to extract, the contour obtained by this method won't be right. That is

why defining a good and appropriate cost function is the essential task of this method. In both techniques, the processes are similar: the first step is the definition of some interesting features (F). A feature is supposed to describe certain properties of the boundary and of its environment (gray levels of the contour, of the background, ...). The next step is the conversion of these features into cost functions (C). A feature can for example be the gradient magnitude of the image, and the associated cost function can be the inverse of the gradient magnitude, giving higher cost to smaller gradients (i.e. weak contrasts) and lower cost to higher gradients (i.e. strong contrasts). The conversion of a feature into a cost function is achieved by a so called cost assignment function (CAF). Figure 3.23 illustrates all these concepts.

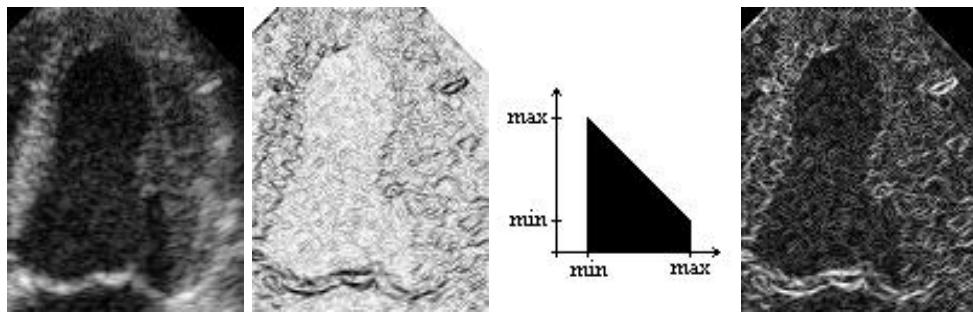


Figure 3.23. Illustration of the cost assignment function: From left to right: the initial image; the feature (gradient magnitude here); the *CAF* (an inversion); the new defined penalty (the inverse of the gradient magnitude).

The following list records some features quoted in the *Live-Wire* [49] and *Intelligent Scissors* [127] papers:

- Gradient magnitude,
- Direction of the gradient magnitude,
- Laplacian,
- Intensity on the positive (inside) side of the boundary,
- Intensity on the negative (outside) side of the boundary,
- Intensity on the boundary (edge).

The cost assignment process depends on how one wants to emphasize one or another value of the feature. For the gradient magnitude the inverse can for example be taken as a *CAF* in order to favor high contrasts, but a Gaussian function, centered on the gradient value one wants to highlight, could also be applied. For the Laplacian feature, the *CAF* is usually a zero-crossing detector. But many other functions may be used according to the feature values to highlight.

Once satisfying individual cost functions are available, the last step consists in combining them into a total cost function. Let us call potential the weighted sum

of all the individual cost functions. This word of potential comes from the Active Contours approach where the energy of the boundary is defined as the integral along this boundary of a functional called potential. On the directed graph-arc from a pixel p to an adjacent pixel q , the potential used by *Intelligent Scissors* [127] is defined by equation:

$$\begin{aligned}\mathcal{P}(p, q) &= \omega_g \mathcal{C}_g(q) + \omega_L \mathcal{C}_L(q) + \omega_d \mathcal{C}_d(p, q) \\ &+ \omega_i \mathcal{C}_i(q) + \omega_o \mathcal{C}_o(q) + \omega_e \mathcal{C}_e(q)\end{aligned}\quad (3.3)$$

where \mathcal{C}_x are the cost functions associated to the features as follows:

- \mathcal{C}_g : gradient feature;
- \mathcal{C}_d : gradient direction feature;
- \mathcal{C}_L : Laplacian feature;
- \mathcal{C}_I : inside intensity feature;
- \mathcal{C}_O : outside intensity feature;
- \mathcal{C}_E : edge intensity feature;

and each ω_x is the weight of the corresponding cost function.

The energy of a path is then defined by

$$E_{\text{path}} = \sum_{(p,q) \in \text{path}} \mathcal{P}(p, q) \quad (3.4)$$

On-The-Fly training

For some features it is hard to decide without prior knowledge about the boundary to extract which values are to be preferred in the cost function. The notion of *On-The-Fly* training is introduced in *Intelligent Scissors* [127] and consists in adapting, during the extraction, the cost functions of the features to the specificities of the contour one wants to segment. It is done for each feature independently from each other. The idea is the following: assuming that the user has drawn a long enough and valid boundary segment, the cost function of the feature has to be modified in order to favor contours with the same feature-values than those found on the segment. An example with the edge intensity feature is shown in table: if the extracted boundary segment is rather dark, the cost function will be modified in order to favor dark intensities. In practice, the process does not modify directly the cost function but the *CAF* : the feature values found on the valid boundary segment (called training path) form an histogram, called training histogram, and the cost *CAF* is iteratively modified by removing from it the training histogram and scaling the result between 0 and 1. Figure 3.24 and 3.25 illustrates respectively the initialization and an iteration of this process. The interest of *On-The-Fly* training is that the potential can be adapted during the extraction, providing the possibility of following a contour with slowly changing properties.

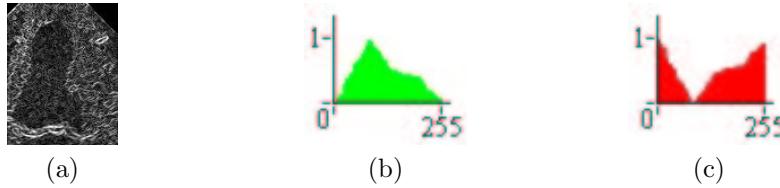


Figure 3.24. Training initialization: (a) the features trained; (b) its corresponding histogram at initialization; (c) its corresponding cost assignment function.

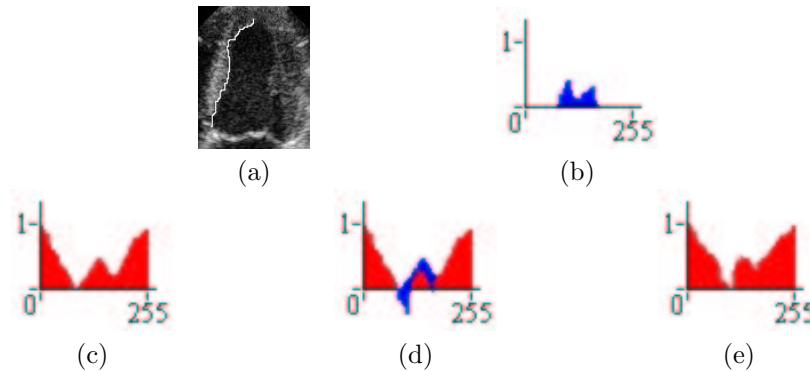


Figure 3.25. Training iteration: (a) the trained path; (b) the histogram along the trained path; (c) the cost assignment function at iteration i ; (d) the CAF minus the trained histogram; (e) the scaled CAF for iteration $i + 1$.

Path-Cooling

With the minimal path approach it is impossible to extract a closed contour with only one seed point and one end point. Indeed, it would mean that two different paths could pass through a same point. To extract a close contour two seed points, at least, are needed. Furthermore, if the extracted path becomes too long, the path search method will prefer shorter paths cutting through the background: it is often necessary to fix several points to draw the expected contour. *Path-Cooling* was introduced in *Intelligent Scissors* [10], as *Border Cooling* and achieves automatic generation of seed points. When a new seed point is generated, the boundary segment between this new point and the previous seed point is fixed (frozen). A new start point is generated when a pixel in the contour is considered to be stable enough. The stability criterion is function of both the time spent on the active boundary segment and the path coalescence (in other terms: how many times a point has been "drawn"). For every pixel in the image two counts are considered: the time history (in milliseconds) counts how long the pixel has been included in the active boundary (boundary section that is not frozen), and the redraw history counts how many times the pixel has been redrawn by the active boundary. When the mouse moves, drawing a new optimal path, the redraw history of the active pixels is incremented while the redraw history of the non-active pixels is set to 0, and the time history of the active pixels is updated

by adding to the current value the while that the boundary segment was displayed and a gradient term (to have a link with the data).

Both histories have two thresholds: a low and a high. When a pixel in the active boundary satisfies the two low thresholds it becomes a candidate point. The first candidate pixel which active boundary segment contains a pixel that satisfies the two high thresholds becomes the new seed point and the rest of the active boundary is frozen. The low thresholds have to be small in order to select candidates as close to the current free point as possible. The high thresholds have to be relatively large to freeze relatively long segments.

3.2.2 Adaptations and improvements

Our work is more based on the *Intelligent Scissors* than on the *Live-Wire*. In this way, we adopt the pixel based graph version: the nodes of the graph are the pixels, and the oriented arcs represent the oriented links between pixels. Our cost functions are directly inspired from the article [127] and we also use *Path-Cooling* and training. The original contribution essentially lies in the introduction of a more general path search, in the adaptation of the cooling speed, and in the way the training is achieved.

Path search algorithms

One of our tasks was to examine the possibility of using the path extraction detailed in section 1.1.2 in the framework of 2D-*Live-Wire* and *Intelligent Scissors*. This extraction method is based on *Cohen and Kimmel* work [34] and uses *Eikonal equation* for propagation. In our implementation we use several path search methods. We used a modified version of *Dijkstra*'s algorithm, which is the basis of the methods developed in *Live-Wire* [49] and *Intelligent Scissors* [127]. We also developed another implementation based on the path search algorithm proposed by *Cohen and Kimmel* [34], already used for virtual endoscopy in the preceding section. We compare the results obtained with both methods.

Modified version of Dijkstra's algorithm On elongated objects, Dijkstra's classical algorithm might perform poorly. This is due to the minimum cumulative cost principle: as the cumulative cost is defined as a sum along the path, the path's cost is a strictly increasing function of the path's length. Thus, the longer the object is, the more likely will the extraction select shorter paths cutting through the background. See an example in figure 3.26. To overcome this problem, we can use a different expression of the cumulative cost in the Dijkstra's algorithm (see [54]), which allows longer paths, by means of introducing recursivity in the cumulative costs computation (see section 2.5 for details).

Eikonal formulation The main idea of *Cohen and Kimmel* [34] is that the potential and the graph are considered to be continuous, producing a sub-pixel path. With this approach, detailed in chapter 1, the energy to minimize is defined as the integral of a strictly positive functional having low values close to the desired features (see equation (1.3)).

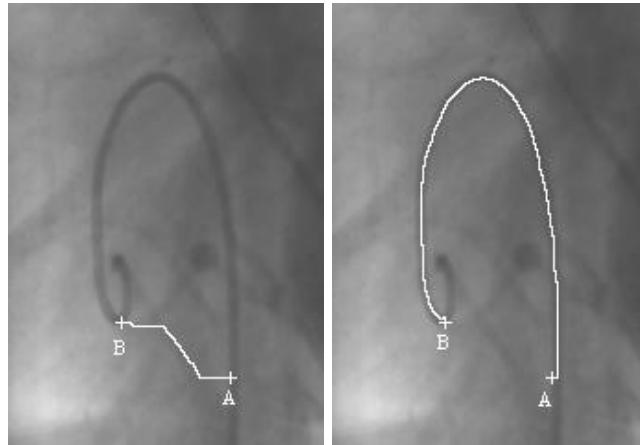


Figure 3.26. Guide-wire extraction in a X-Ray image: Left image - Path extraction with the classical Dijkstra's algorithm between points A and B; right image - Same path extraction with improved Dijkstra's algorithm.

Comparison between Dijkstra and Eikonal equation

As explained in the first chapter, the main difference between *Dijkstra* and *Cohen and Kimmel* definition of the minimal path lies in the considered metric. In the first case, the minimal path is the one where the sum of the potential is minimal (L_1 path), and in the second case, it is the one where the integration of the potential is minimal (L_2 path). With *Dijkstra*'s approach, the image is considered as a graph in which each pixel is a node and the weights on the vertices are functions of the energy to minimize. This method uses dynamic programming to compute the optimal path [43]. *Cohen and Kimmel* approach keeps a continuous framework for the problem and computes the path by solving the Eikonal propagation equation in real-time with a Fast Marching algorithm. The aim of the presented work is to achieve real-time extraction. Even if Eikonal method uses integrals to compute the optimal contour, it is not very slower than *Dijkstra*'s approach that uses sums. For the guide-wire extraction shown in figure 3.26, the computing time ration between both methods is about 0.89. And, because *Eikonal equation* produces a sub-pixel path (L_2 path), it is more accurate. The continuous formulation of *Cohen and Kimmel* [34] method has the advantage to keep a more general framework for the energy definition, allowing for example applications using other kind of potentials. It also easily include an offset term w (see equation (1.3)) to constrain the regularity of the path, while it is more difficult with dynamic programming methods [117, 62].

3.2.3 Dedicated potential

To build the potential (i.e the weighted sum of cost functions), two cases are distinguished: the object to extract is either an interface between two regions, or a line (ridge) over a uniform background. The line approach was motivated by the

patents [55, 56], which are based on a ridge filter to proceed to the extraction of linear structures. It was therefore possible to compare quickly the two path search methods (discrete and continuous), and have a rapid overlook over the facilities of the *Live-Wire* and *Intelligent Scissors* tools. For the region interface approach, we use the six features quoted previously: the gradient magnitude, the Laplacian zero crossing, the gradient direction, the edge intensity, the inside intensity and the outside intensity, as described in [127]. Thus, the potential at an oriented arc (p,q) is described by the equation (3.3). In the next section, the defined costs functions have values scaled between 0 and 1. To display the cost maps we re-scale the values between 0 and 255 in this way: if C is the cost function and MC is the cost map associated to C, at a pixel (i,j).

Specific case: long curve extraction

A ridge-filter potential is used for line extraction and is based on local contrast estimation, seen as the difference between a tangential and an orthogonal term, relatively to the direction of the line to extract. It is similar to method described in [96]. Discrete and continuous implementations can be found in [109, 60]. For more details, see [54, 55, 56].

General case: Contour extraction

The potential used for the general case of contour extraction is the one introduced in equation (3.3).

- Gradient magnitude: this feature is useful to locate contrasted areas. As a first order derivative operator, it is a representation of the spatial variations of the image. The gradient of an image $I(x,y)$ is defined by: $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$ and taking $G_m = \sqrt{I_x^2 + I_y^2}$ the cost function is given by

$$\mathcal{C}_g(x, y) = \frac{\max_I G_m - G_m(x, y)}{\max_I G_m - \min_I G_m} ; 0 \leq \mathcal{C}_g \leq 1$$

The image is convolved with a Gaussian kernel, before computations.

- the Laplacian zero crossing: As a second order derivative operator, the purpose of the Laplacian zero-crossing is edge localization. The Laplacian of an image I is defined by $L(I) = I_{xx} + I_{yy}$. The corresponding cost function to the Laplacian feature is the Laplacian zero-crossing (LZC). In theory it is defined like this: the LZC is 0 where the Laplacian is 0 and 1 everywhere else. But in practice such a LZC does not produce many zero-crossing points. That is why the zero-crossing area is extended to the pixels where the Laplacian changes its sign with a specific gap. The gap has an influence on the strength of the contrast one wants to highlight with the cost function: the deeper the gap is, the stronger the selected contrast will be. See figure 3.27. Therefore, a zero-crossing is defined by two points with Laplacian of opposite signs. The point with its Laplacian

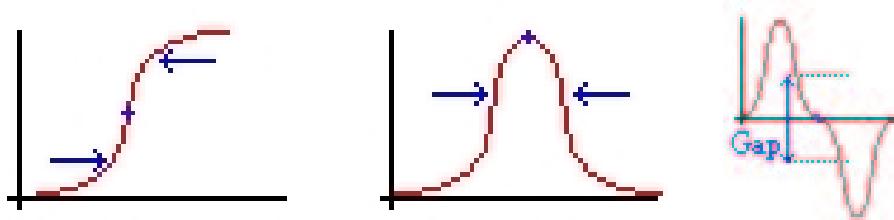


Figure 3.27. Influence of the depth of the gap: left image - profile of an image intensity; middle image - profile of the associated gradient; right image - profile of the associated Laplacian.

closer to zero than the other is set to be the *LZC*. Actually, a pixel is a *LZC* if its Laplacian is zero or:

- First, it is closer to zero than any of its neighbors with a Laplacian from opposite sign,
- Then, there is at least one opposite sign neighbor so that the gap between them is sufficient.

Consequently, the *LZC* map is a binary map defined by:

$$\begin{cases} \mathcal{C}_L(p) = 0, & \text{if } L(p) = 0 \\ \mathcal{C}_L(p) = 0, & \text{if } \exists q \in N(p) \setminus [(L(p).L(q) < 0) \cap (|L(p)| < |L(q)|) \cap (|L(p) - L(q)| < \text{gap})] \\ \mathcal{C}_L(p) = 1 & \text{otherwise} \end{cases}$$

where $N(p)$ is the neighborhood of p .

- The gradient direction: introduces a smoothness constraint into the potential. The associated cost function is not local (i.e. defined on one pixel) but measures the continuity of the gradient direction between two adjacent pixels. We use the formulation as defined in *Intelligent Scissors* [127], for the gradient direction cost from pixel p to pixel q :

$$\mathcal{C}_d(p, q) = \frac{2}{3\pi} \{ \arccos[d_p(p, q)] + \arccos[d_q(p, q)] \} ; 0 \leq \mathcal{C}_d \leq 1$$

where $\arccos d_p$ and $\arccos d_q$ represent the angles between the link (p, q) and the gradient direction in respectively p and q . d_p and d_q are computed with

$$\begin{aligned} d_p(p, q) &= D'(p).V(p, q) \\ d_q(p, q) &= V(p, q).D'(q) \end{aligned}$$

with

$$\begin{aligned} D'(p) &= \frac{1}{\sqrt{I_x^2 + I_y^2}} (I_y(p), -I_x(p)) \\ V(p, q) &= \frac{1}{\|p - q\|} \begin{cases} q - p, & \text{if } D'(p).(q - p) \geq 0 \\ p - q, & \text{otherwise} \end{cases} \end{aligned}$$

This potential associates a low cost with an edge between two adjacent pixels where the gradient of the pixels and the link between them have similar directions. Furthermore, it associates a high cost with an edge between two adjacent pixels that have similar gradient directions but are almost perpendicular to the link between them. In practice, the efficacy of such a cost is not obvious and we do not use it in the potential. Applying a smoothing operator to the extracted curve after the extraction would probably have a better effect.

- The pixel intensities: The edge intensity is given by the scaled value of the source image at the boundary; the inside intensity is obtained at some offset k from the boundary in the gradient direction and the outside intensity comes from the image intensity at the same offset k from the boundary in the opposite of the gradient direction. Calling \mathcal{C}_e , \mathcal{C}_i and \mathcal{C}_o respectively the edge, inside and outside feature costs, their formulations are:

$$\begin{aligned}\mathcal{C}_e(p) &= \frac{1}{255} \mathcal{I}(p) \\ \mathcal{C}_i(p) &= \frac{1}{255} \mathcal{I}(p + k \frac{\nabla \mathcal{I}}{\|\nabla \mathcal{I}\|}(p)) \\ \mathcal{C}_o(p) &= \frac{1}{255} \mathcal{I}(p - k \frac{\nabla \mathcal{I}}{\|\nabla \mathcal{I}\|}(p))\end{aligned}$$

Without training these features are not used in the potential because it is impossible to decide without prior knowledge about the contour to be extracted which values are to be preferred. The aim of training is to associate with them an adequate *CAF*.

3.2.4 Path-Cooling improvement

We tested two different *Path-Cooling* methods. Both are based on the same idea: if a point in the active contour is stable enough, the corresponding boundary segment is frozen. The first process consists in having one counter, the redraw history, and one threshold. The other approach uses the two counters described in [127], with an adaptation: the time history is updated by multiplying (and not adding) a scaled potential-driven factor instead of a simple gradient-driven factor. The multiplication has a weighting effect which gives more influence to pixels with a low potential. At a pixel p and an iteration i of the cooling process, the time history \mathcal{TH}_i is defined as

$$\mathcal{TH}_i(p) = \mathcal{TH}_{i-1} + t_{i-1} \times [1 - \mathcal{P}(p)]$$

where t_i is the consecutive time the path was active until iteration i and \mathcal{P} the penalty in equation (3.3).

The time history is useful if we consider that when the user does not move (redraw history fix, but time history incremented), he wants to emphasize the already drawn contour. But, as both thresholds are to be satisfied, even if the user does move very slow, or does not move at all, the active path will freeze slowly. In practice, the definition of the thresholds is not obvious and the main difficulty lies in the

interpretation of the mouse movement, in terms of speed and acceleration. We can either increase the cooling speed with the mouse cursor speed or decrease it. An argument to increase the mouse speed is the following: if the user moves fast it is because there is no real difficulty with the drawing and because he considers the extracted path as valid. But if the cooling speed is proportional to the mouse speed, and if the user has to define little areas where the contour has lots of details, he must fix a manual seed point: in these areas he must go slow and the cooling process will also go slower. An argument to decrease the cooling speed is the following: the areas where the user goes slowly are the areas which are not very well defined, where the path changes quickly its aspect... , deciding then that the slower the mouse moves the quicker the user wants the path to freeze. The problem is that if the user goes too slowly in other areas (hesitating, for example), he may fix false paths. Even if the second option, with practice and taking care to remain close to the boundary, seems to be the most adequate, no choice is totally satisfying in a general case. However, the *Path-Cooling* is a very satisfying tool to extract closed boundaries, as shown in figure 3.28.

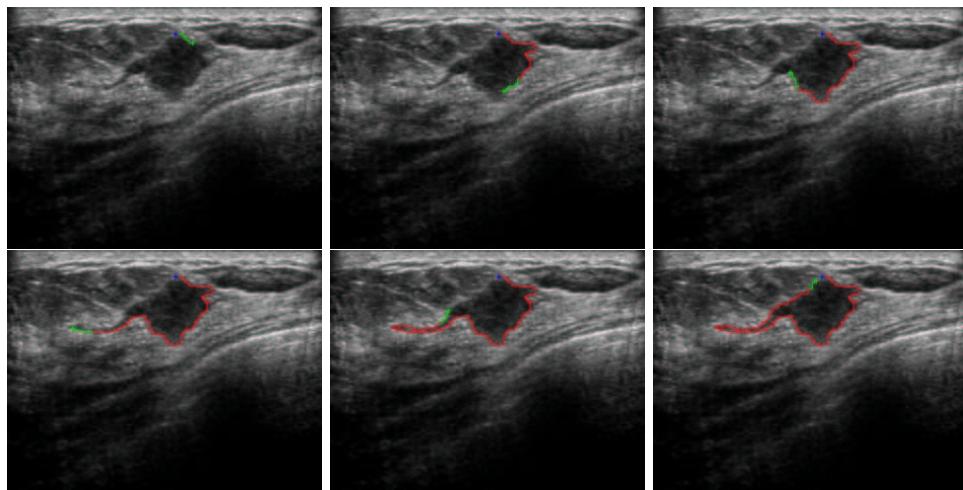


Figure 3.28. *Path-Cooling* on an ultrasound mammography image: These are iterations of the delineation of a tumor; the blue cross is the see point at iteration 0; the green cross is the mouse cursor; the red path is the “cooled” one; the green path is the currently drawn path between the mouse cursor and the seed point at each iteration.

3.2.5 On-The-Fly training improvement

In the used potential (see equation (3.3)), training can be applied to the gradient magnitude \mathcal{C}_g , inside \mathcal{C}_i , outside \mathcal{C}_o and edge features costs \mathcal{C}_e .

Training path

Training, as described in [127], is based on the distribution of the feature values on a valid contour segment. The signification of valid is not obvious. We tested three approaches to define such a segment (See figure 3.29):

1. the training path is the last section of the frozen contour and training is applied at each setting of a seed point (blue points in figure 3.29);
2. the training path is the last section of the active boundary and training is applied at each mouse movement, i.e. at each path extraction (green points in figure 3.29);
3. the training path is linked to the cursor position: it corresponds to the points where a mouse movement event is sent to the system, and training is applied at each mouse movement (red points in figure 3.29).

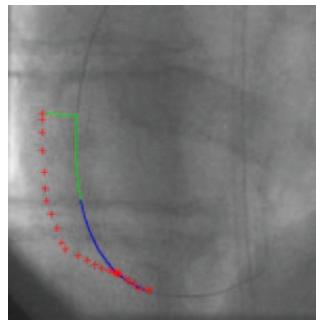


Figure 3.29. Several training paths: On this fluoroscopy image, the cursor position is represented by red crosses, the frozen part of the path is in blue, and the current free part of the path is in green.

Training is effective when a new seed point is manually or automatically set. This setting of a seed point can be seen as the validation of a path segment. The free path, because of its high variability and dependence to the potential (and thus on the training), cannot be a suitable training area. As well as for the mouse movement marker, because it would make it impossible to go too far away from the contour with the mouse cursor.

New way of training

We have developed an improvement of the classical training method (see patent [71]). In existing methods the training area is limited to a portion of the path itself and only takes a positive information into account (see section 3.2.1). The original technique we use is based on the addition of another training area based on negative information. The positive training area contains pixels that could belong to the contour, while the negative training area contains pixels that do not belong to the contour. The positive area has a reinforcing role while the negative area has a penalizing role in

the cost assignment process. This improvement has two consequences: firstly, the new potential is more robust and secondly, the comparison of the distributions of the features (i.e. of the histograms) on each area helps adapting the weights of the individual cost functions in the total potential.

Definition of the positive and negative training areas The main problem is to define suitable positive and negative areas that describe well enough respectively what is and what is not a contour pixel. The definitions we use are all based on the previously described training path. As the drawing of the user is not very accurate, we consider the positive area in the neighborhood of the training path and we tested four approaches for the negative area definition:

1. in the minimal box including the training path, the points closer to the path than a certain distance d are considered to be the positive area, the other points of the box are considered to be the negative area (see figure 3.30-(a));
2. in the minimal box including the training path, the points closer to the path than a certain distance d_p are considered to be the positive area and the points further from the path than the distance d_p and closer to the path than a certain distance d_n are considered to be the negative area (see figure 3.30-(b));
3. the positive and negative areas are made from paths coming from the neighborhood of the click-position. The paths coming from the nearest neighborhood form the positive area and the path coming from the furthest neighborhood form the negative area (see figure 3.30-(c));
4. The training set of points is made from translations of the path: in the minimal box including the training path, the nearest translations are the positive area and the furthest translations are the negative area (see figure 3.30-(d));

For each approach, it is possible to add a weighting function based on the distance to the training path.

The first approach is not accurate enough for the definition of the negative area. Indeed it is possible that other points of the box belong to another right path section. It is the motivation for introducing the second approach. The third method seemed a priori to be the most accurate to define a negative area. But actually, all the paths are rapidly concurrent, what on the first hand misapprehends the notion of positive/negative training point, and on the other hand limits the size of the training set. Second and last approaches are very similar and give the best results with this difference that the first one is a continuous version of the second one. We therefore choose the last approach: the positive area is the set of p nearest translations and the negative area is the set of n next translations of the training path. The translation direction is chosen perpendicular to the mean direction of the training path. The different translations are weighted according to their distance to the training path (see figure 3.31-(d)). The negative/positive areas are symmetric for the gradient magnitude and the edge intensity features (figure 3.31-(a)). But, in order to consider the non symmetrical aspect of the inside and outside features (in equation (3.3)), we adopt for them non symmetric training areas, depending on the direction of the

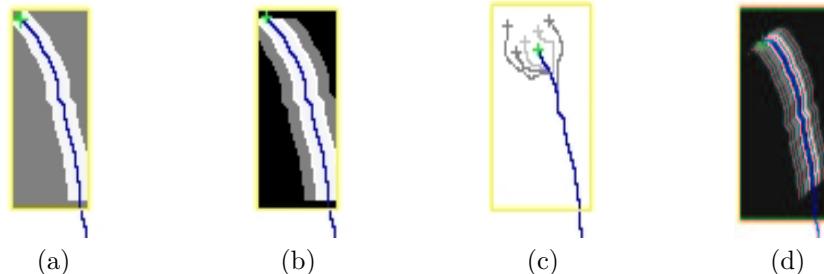


Figure 3.30. Different definitions of the positive and negative areas: Positive area in white, negative area in gray, other points of the box in black. (a) Positive area is a lane around the box, negative area is the rest of the box. (b) Positive area is a lane around the box, negative area is a lane around the positive area. (c) Areas built by the paths coming from a region around the mouse cursor. (d) Areas built with translations of the training path.

gradient on the path, the path direction and the considered feature. See examples with figure 3.31-(b)-(c). The positive/negative training sets of points are used to build



Figure 3.31. Training areas: In white the positive area and in black the negative one. The longest line is the training path. (a) Symmetric. (b) Asymmetric for inside feature. (c) Asymmetric for outside feature. (d) Schematic weighting function.

two distinct histograms of the features values. The function of these histograms is twofold: to build an adapted cost function for the feature (through the construction of an adapted *CAF*) and to adapt automatically the weight of the individual cost function in the global potential.

On-The-Fly adaptation of individual cost functions Individual cost functions are built with a *CAF* applied to the feature. Training is used to dynamically modify this cost assignment function. With our method, the algebraic difference between the positive and the negative histograms (jointly scaled) is removed from the iteratively modified *CAF* and the result is normalized to compose a new cost.

The positive and negative histograms are scaled the following way: firstly, to have a same scale for both training histograms, the negative histogram values are multiplied by the ratio between the number of points used to build the positive histogram and the number of points used to build the negative histogram:

$$H^-[i] = \frac{\text{card}\{H^+\}}{\text{card}\{H^-\}} \times H^-[i]$$

where H^+ and H^- are respectively positive and negative histograms. Then, both histograms are normalized using their common min/max. The initialization used in [127] favors the gray values with highest occurrence in the feature. This is incorrect initialization for images where there is a large and homogeneous background as in figure 3.32-(a). We prefer an approach that does not carry any a priori about the expected feature values: the CAF is initialized with a flat line, giving the same cost to each pixel of the image, and not with the inverse of the distribution of the feature. As a consequence, the CAF obtained during the process depends only on the past and the present training data.

Computing a cost function using a positive and a negative information into account makes the method more robust. Actually, the positive training area describes what is a contour and the negative training area describes what is not a contour. So if the training contour is not enough uniform, producing a too wide positive histogram, the classical approach will favor points that are perhaps not relevant to the expected contour, whereas our method uses the negative information to localize the contour, producing a less specific but more accurate cost function.

We show an example of *On-The-Fly* training on the septum wall of an echographic left-ventricle image in figure 3.32.

Figure 3.32 illustrates the first iteration of the process of training on the septum wall of an echographic left-ventricle image (figure 3.32-(a)) with both approaches. The trained feature is the inside intensity one (figure 3.32-(b)). The classical method uses a CAF initialized with the inverse of the distribution of the feature, where black levels are predominant and thus favored (figure 3.32-(c)). The training histogram is scaled between 0 and 1 (figure 3.32-(d)) and removed from the initial CAF to build a new CAF (figure 3.32-(e)) favoring very specific values. With this example the training path is homogeneous and the resulting cost function (figure 3.32-(f)) is good. But with a not uniform enough training contour, the resulting potential will not be consistent. Our method initializes the CAF (figure 3.32-(g)) with an arbitrary value between 0 and 1. The positive and negative histograms (figure 3.32-(h) and figure 3.32-(i)) are jointly scaled and the difference between them is removed from the initial CAF producing a less specific but carrying more accurate information histogram. Thus the new cost function (figure 3.32-(k)) is more relevant.

A real case study of the *On-The-Fly* adaptation of the individual cost functions is shown in figure 3.33, where iterations of the modification of the CAF of the feature \mathcal{C}_e are shown.

Adaptation of the weights The principal advantage of using positive and negative training areas is the evaluation of the differences between the histograms, which helps adapting the weight of the corresponding individual cost function in the global potential. If the histograms are enough distinct, we can assume that the considered feature is enough discriminating and that its weight should be more important. In a first approach we take a mean difference between both histograms as dissimilarity criteria, which expression is the following:

$$c = \frac{1}{256} \sum_{i=0}^{255} |H^+[i] - H^-[i]|$$

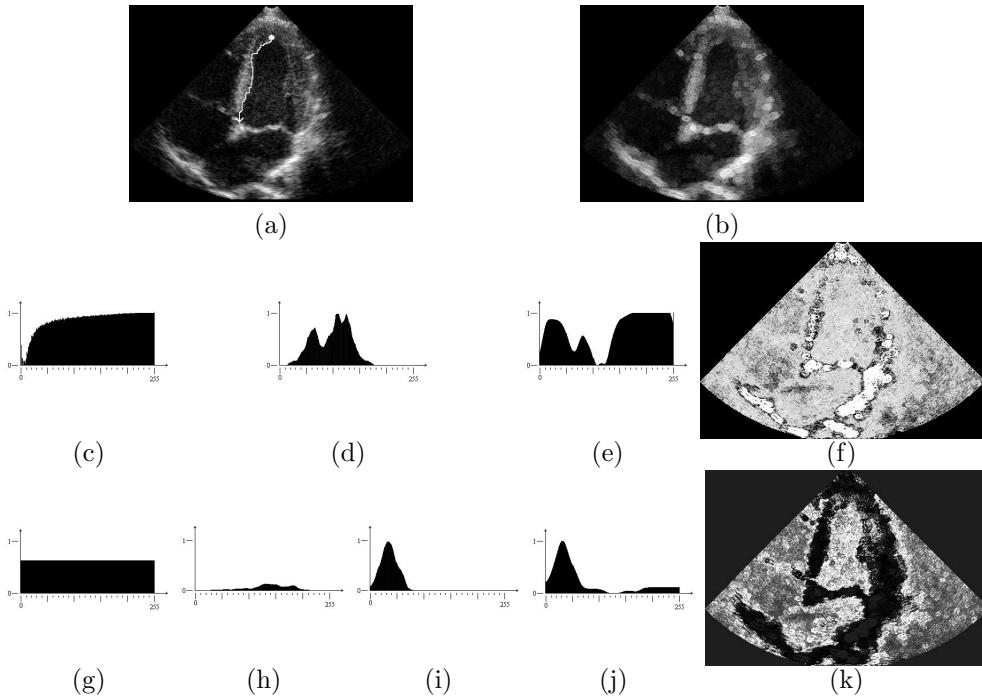


Figure 3.32. Training on the septum wall of an echographic left-ventricle image: (a) echographic left-ventricle image; (b) inside Feature C_i ; (c to f) **classical training**: (c) initial CAF ; (d) training histogram; (e) resulting CAF with (f) corresponding cost function; (g to k) **improved training**: (g) initial CAF ; (h) positive and (i) negative training histograms; (j) resulting CAF with (k) corresponding cost function.

Indeed, if the histograms are very similar this criterion will be very low and if they are different, it will be high. In practice, this criterion produces often very low values and quite never values above 0.5. Hence we use a stretching function to exploit efficiently this criterion and transform it into a weight ω_c associated with the cost function c in the total potential. See on figure 3.34 examples of stretching functions.

3.2.6 Conclusion

We have developed an interactive, real-time and user-guided image segmentation software, which gives to a non-specialist the possibility to outline the contour of an object in an image without a very precise drawing (for example with the track-ball on an echograph). Figure 3.35 displays several example of the interactive drawing tool for medical images. Some improvements have been brought to the bibliographical background of interactive extraction of optimal path. A very general optimal path extraction method has been efficiently used, producing in real-time very precise paths. And a new method of *On-The-Fly* training has been developed to adapt, during the

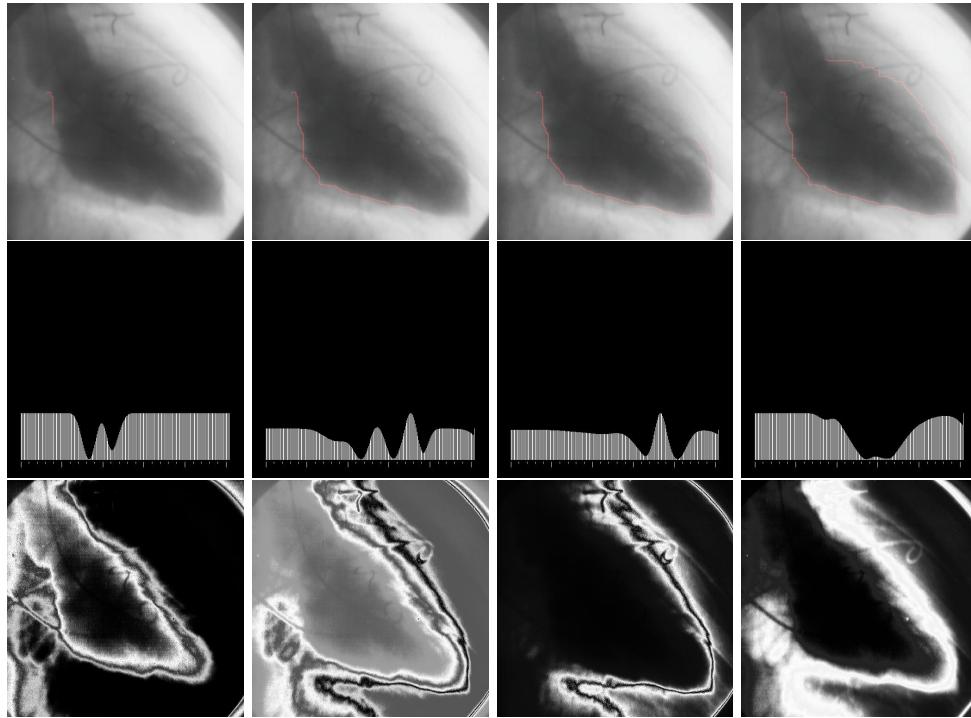


Figure 3.33. Results of the training on a left ventricle image: First row - iterations of the real-time contour extraction with training, on a X-Ray image of the heart left-ventricle; second row - modification of the CAF of the feature C_e at the same iterations; third row - corresponding feature C_e potential at the same iterations.

extraction of the contour, the individual cost-functions and their relative weights in the total potential.

3.2.7 Perspective

- Training:
 1. creating a better dissimilarity criterion between information from positive and negative histograms;
 2. ordering the importance of each different computed criteria with the training facility, knowing at each iteration what feature is really discriminant;
 3. use such a method to select *off-line* interesting features into a large database of them (like the size of the Gaussian kernel used to smooth the image).
- *Path-Cooling* : The acceleration of the mouse cursor could be an interesting information for freezing trajectories;

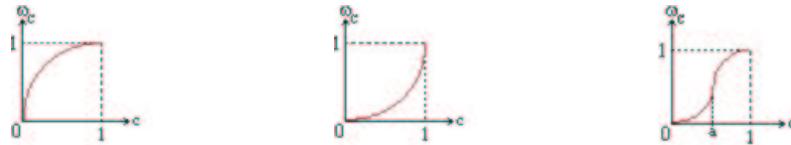


Figure 3.34. Examples of stretching functions: (a) Privileges small values of the cost; (b) penalizes small values of the cost; (c) penalizes values of the cost below a and favors values of the cost above a .

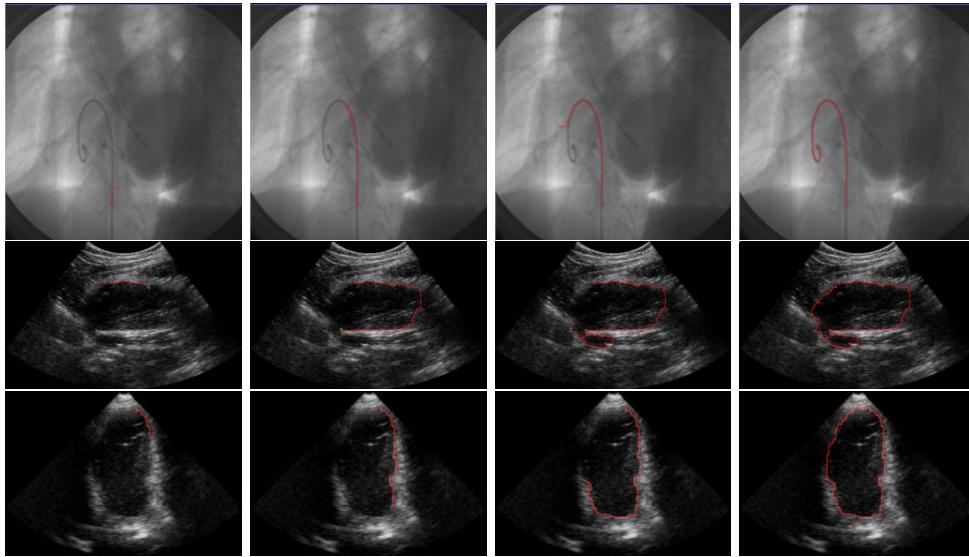


Figure 3.35. Results on different datasets: First row - Extraction of a guide-wire in a fluoroscopy image; second row - tumor delineation in a ultrasound mammography image; third row - extraction of the left ventricle in an ultrasound image.

- 3D extension: using a 3D path search method (as in section 2.1), or perhaps a "surface-search", method instead of a slice-by-slice approach ([46, 47]). However, with a slice-by-slice method, a contour extracted with our technique could be an interesting initialization for other pure 3D approaches (as simplex meshes [38] for example).
- Interactivity: grading the level of interactivity, and limit the user interaction to the choice of this level.

